

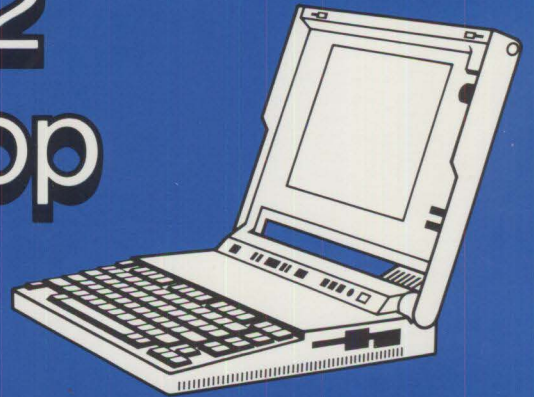
Issue 3, 1991

Personal Systems

Featuring
LANs

IBM Operating System/2[®]

PS/2
Laptop



IBM Personal Systems Technical Solutions



Trademarks

IBM, AIX, Micro Channel, Operating System/2, OS/2, Personal Computer AT, AT, Personal System/2, and PS/2 are registered trademarks of International Business Machines Corp.

Audio Visual Connection, AVC, Communications Manager, Database Manager, DB2, Hollywood, LinkWay, Personal Computer XT, PC XT, Presentation Manager, RISC System/6000, SQL, SQL/DS, Storyboard Live!, Systems Application Architecture, and SAA are trademarks of International Business Machines Corp.

AT&T is a registered trademark of AT&T.

AutoCAD is a registered trademark of Autodesk Inc.

Hayes is a registered trademark of Hayes Microcomputer Products.

Intel is a registered trademark, and i860, 386, i486, 486, i486sx, 80286, 80386, 80386SX, 80486 and ActionMedia are trademarks of Intel Corp.

Lotus 1-2-3 is a trademark of Lotus Development Corp.

Macintosh is a registered trademark of Apple Computer Inc.

Metaphor is a trademark of Metaphor Computer Systems

Microsoft and MS-DOS are registered trademarks, and Windows and CodeView are trademarks of Microsoft Corp.

MNP is a trademark of Micro Com.

Novell and NetWare are registered trademarks of Novell Inc.

OSF and Open Software Foundation are trademarks of Open Software Foundation Inc.

UNIX is a registered trademark of UNIX Systems Laboratories Inc.

Xerox and Ethernet are registered trademarks of Xerox Inc.

PS/2 Model L40 SX Laptop Portable Computer

*Leo Suarez, Frank Canova, and Neil Katz
IBM Corporation
Boca Raton, Florida*

Portable computers are not new to the industry, and until recently have been plagued with many limitations, such as size, weight, usability, and enough horsepower to match their desktop counterparts. With rapid advances in low-power logic, small lightweight DASD, and advances in flat-panel technology, the time was right to introduce the PS/2 Model L40 SX Laptop Portable Computer, a unit that matches its desktop counterparts.

Market studies have shown that customers are looking for powerful, Intel 80386SX-based machines, with a minimum of 40 MB of hardfile capacity, a 1.44 MB diskette drive, and enough onboard memory to run large, memory-intensive applications such as OS/2 or Windows®. Customers also require that the machine's keyboard and screen be comfortable to use.

Another requirement is that machines are battery-operated for true portability. Another is expansion capability – the ability to add communication cards, additional DASD, or a SCSI card. Finally, customers ex-



pect all these features in a small lightweight package that can easily fit inside a briefcase.

Hardware Overview

The PS/2 Model L40 SX is a high-performance Intel® 80386SX™, 20 MHz-based laptop portable computer. It is designed around a chip set that has been optimized to reduce the total number of external devices required to implement a complete IBM AT®-bus (non-Micro Channel®) computer.

The L40 SX comes standard with 2 MB of dynamic random access memory (DRAM) expandable up to 18 MB via two single in-line memory module (SIMM) sockets. The standard storage devices in the L40 SX are a 3.5-inch, 2 MB (1.44 MB for-

matted) superslim floppy diskette drive, and a 60 MB, 2.5-inch hard disk drive with 19 millisecond (ms) average seek time.

Integrated into the machine is a 640 by 480, VGA-compatible, monochrome, side-lit, super twisted nematic liquid crystal display (STN LCD) with 32 addressable grey scales. A full-function keyboard is also integrated into the base machine. Included is a separate numeric keypad that, when positioned next to the computer, provides the user with a complete 101-function keyboard.

The ports include a 9-pin serial port, 25-pin parallel port, external VGA monitor port, a combination mouse or numeric keypad port, and an ex-

ternal expansion port for docking-station connectivity. Internal plug-in options for the L40 SX include 2 MB, 4 MB, or 8 MB SIMMs that plug into two SIMM sockets.

Also, a local internal bus provides the interface required for either an optional RS-232D serial adapter or a 2400 bps Hayes®-compatible modem with an integrated 9600 bps fax. The entire system is powered by either a 40 watt external AC/DC adapter or an internal 26 watt-hours (W-Hr) nickel cadmium rechargeable battery. With a second, smaller nickel cadmium battery, users can remove the main battery without shutting off the computer. With this feature, users can plug in a fresh battery or AC/DC adapter if the main battery runs out of power.

The L40 SX power management system allows users to turn off unused subsystems such as the diskette drive or external ports. This, in turn, optimizes battery life for individual applications. The power management system is powerful enough to shut down sections of logic between keystrokes.

Other options available include:

- A battery quick charger, which recharges batteries in less than 2.5 hours
- An automobile adapter, which operates the L40 SX from a cigarette-lighter socket
- A mouse/trackball device, which integrates the features of a mouse and trackball into a small package
- A premium carrying case with room for adapters, diskettes, and notebooks

The entire system is contained in a 12.8-by-10.7-by-2.1-inch package weighing only 7.7 pounds.

Description

System Board: The model L40 SX system board contains the logic necessary to support a fully compatible IBM PC. The system board includes the system processor, a coprocessor socket, 2 MB of DRAM, AT bus control logic, video control logic, video RAM, external I/O connectors, and a power supply. All the logic is contained in a system board measuring 10 inches by 12 inches. The system board is divided into five sections, as shown in Figure 1.

The first section contains the processor, core logic, and all DRAM. The core logic consists of three highly integrated VLSI chips. The largest of the chips, which contains the system processor control, memory control, and power management logic, contains nearly 30,000 gates. The second core logic chip contains two serial ports and a single parallel port, while the third core logic chip contains a floppy disk controller, system realtime clock, and Integrated Drive Electronics (IDE) for the hardfile interface. All chips are designed with a power control logic feature. This feature allows: the device to be clocked down to 0 MHz; portions of the device to be independently powered down; or the device to be totally powered down.

The second system board section contains the video control circuits. This logic consists of a single VGA controller chip that can drive either the system internal LCD display or an external VGA-compatible CRT. The video controller chip handles all display buffer management functions, including display refresh cycles, memory refresh cycles, and the arbitration of host access cycles. Included with the video controller chip is an integrated random access memory digital-to-analog converter

(RAMDAC). The RAMDAC has a 256-by-18-bit color lookup table with triple six-bit, digital-to-analog (D/A) converters that provide the three analog signals (red, green, and blue) to the video display connector. The video controller contains the logic that maps colors to 32 grey scales for display on the system's LCD.

The third section contains the system I/O connectors. These connectors consist of the standard PS/2 connectors – a single external serial port, parallel port, video connector, AT bus expansion connector, power, and external numeric keypad or mouse connector. The many connectors on the system board connect the system board and the various internal system devices. The internal devices connected are the LCD panel, floppy diskette drive (FDD), hard disk drive (HDD), SIMMs, internal data/fax modem or second serial port, internal keyboard, and battery.

The fourth section contains the user input subsystem (UIS). The UIS allows the computer to monitor the system in which it is operating. UIS supports numerous interfaces, including one that allows the system to accept user input from the keyboard and keypad, and for the system to sense its own environment. UIS functions are controlled by a second microprocessor unit, called the Slave Micro-Control Unit (SMCU). The SMCU performs the scanning function for the internal keyboard. It also supports the external numeric keypad or mouse. As the SMCU is driving the keyboard scanning matrix, it simultaneously performs a number of other functions. The SMCU monitors the battery voltage and temperature, and determines whether to turn on the battery-charging circuits and to execute an algorithm that determines the amount of energy remain-

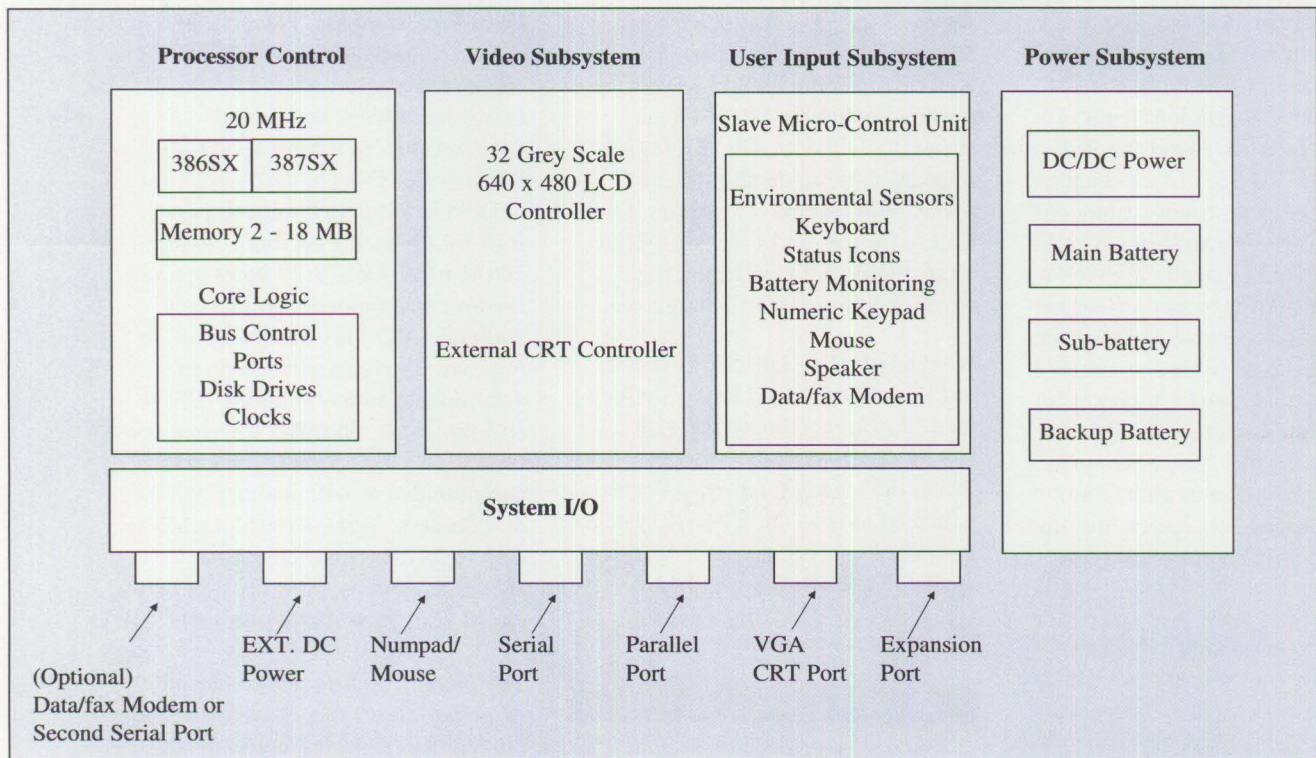


Figure 1. L40 SX System Board Components

ing in the battery. The SMCU also controls the system icon status indicator panel. The SMCU also reads the temperature via its onboard A/D converters, and its humidity sensors determine if the machine's internal environmental conditions are operating within range.

The last section is the DC/DC power supply, which provides all DC power to the system logic, modem, and LCD display. The power subsystem is a simple and efficient (85 percent) bucking regulator that receives its power from either the internal battery pack or external power such as the system AC adapter. Total internal maximum power consumption is 40 watts.

During AC operation, the internal system battery pack is recharged, independent of system operation. The power subsystem is designed to en-

sure uninterruptible power in case of failure. If AC power is lost, the main battery pack takes over, allowing system operation to continue.

There is a small sub-battery within the power subsystem. This battery provides system power when the main battery is being replaced. The sub-battery is continuously recharged, allowing for multiple standby operations.

Hard Disk Drive: The L40 SX uses a 2.5-inch, low-power, low-weight, hard disk drive. The drive has an integrated onboard AT controller that interfaces directly with the WD76C21, which has 16-bit Integrated Drive Electronics incorporated in the chip.

The drive has been optimized for portable applications: low power, small footprint, and low weight. Key

attributes of this hard disk drive are 60 MB minimum formatted capacity and typical 19 ms average seek time.

The hard disk drive has unique power management features that allow the system to power down sections of the drive such as the motor, logic, or both. This is critical, because the machine must minimize power draw if its subsystems are not used. To power down the drive, specific registers on the drive are addressed that power off the logic, motor, or both.

Diskette Drive: Similar to the hard disk drive's criteria for weight, size, and low power, the floppy diskette drive also had the same limitations. The the only exception is that it had to be wider to accommodate the 3.5-inch medium.

Floppy diskette drive technology, however, has not evolved like that of hard disk drives with their on-board power management features. Therefore, L40 SX power management on the diskette drive is handled by the diskette controller features of the WD76C21 device. Whenever devices monitor no diskette activity, a timer counts down from user-set values or the default value is set by the system. If no activity is encountered during this time, the +5 V power to the diskette is powered off. Upon detecting any activity, power is again supplied to the diskette drive and read/write access is allowed after the appropriate spin-up time has been achieved.

Liquid Crystal Display (LCD):

The L40 SX uses a super twisted nematic LCD with a color compensation film to get a black-on-white screen. This compensation film also produces a high contrast ratio (20:1 maximum, 12:1 typical). The display is a transmissive type with a high-efficiency cold cathode fluorescent lamp (CCFL) side light.


The display has 640-by-480 pixels of resolution (VGA). Each pixel is 0.31 mm by 0.31 mm and generates a screen measuring 10 inches diagonally. The display can be divided into four major sections:

- Liquid crystal glass assembly
- Row and column drivers that connect to the glass via flexible cables
- Side-light and diffuser
- Power inverter that provides power to the CCFL

The side-light technique allows the display thickness to be kept to a minimum (10 mm maximum), allowing the overall machine thickness to be reduced. The high-efficiency CCFL also allows high illumination for low

power. Typical power dissipation at 50 candelas/meter square is 3.5 watts. This is the total power of the display, including the CCFL, inverter, and drivers. The display has onboard logic to display 16 grey scales with Gamma correction. However, with the use of the WD90C20 VGA Flat Panel Controller, the system can address up to 32 grey scales.

Keyboard: The L40 SX keyboard has 84 rubber-domed keys. The keyboard layout is compatible with IBM's G-101 keyboard layout. When the standard numeric keypad is placed next to the L40 SX system unit, the keyboard layout is identical to that of the IBM G Keyboard.



The fax utility lets users transmit, receive, or view a faxed image, which can be shown on the system display or sent to one of the printers

A basic requirement was to make typing on the keyboard easy. This meant that the travel between keys, typing angle, and keyboard height must be compatible with existing IBM keyboards. The keyboard typing angle was designed to be five degrees, travel was set at 3 mm, and the height at the home row was designed at 30 mm. These factors made the L40 SX system slightly larger than competing laptops. It also made the L40 SX significantly more usable as a data entry device than competitive models.

Data/Fax Modem: The portable system's design must also allow for connectivity to a variety of data and image facilities. The internal data/fax modem for the PS/2 L40 SX provides this connectivity. This modem is a highly compact, two-inch-by-six-inch card that contains all the logic for a V.29 9600-bps fax send/receive modem, as well as a 2400 bps, V.22 bis data modem. To improve operation efficiency, the data modem supports the MNP™ 4 & 5 protocols, allowing for error correction and data compression when communicating with another MNP 4 & 5 modem. With some types of data, effective data transmission speeds as high as 4800 bps can be realized with these MNP protocols.

The modem is designed to minimize its power draw. When not in use, all modem functions are powered off; turning on Data Terminal Ready (DTR) powers up the modem. The modem also powers up the system if the system is suspended and the modem detects an external ring from a calling device.

Included with the modem is a simple-to-use fax utility. It lets users transmit, receive, or view a faxed image, which can be shown on the system display or sent to one of the printers.

An optional RS-232D card satisfies other countries' widely different requirements for modem/fax functions. This card plugs into the same local bus as the modem/fax card and allows external, country-specific modems to be attached to the L40 SX. With this card, users can add a second serial port to the machine if a modem/fax feature is not required.

Power Management

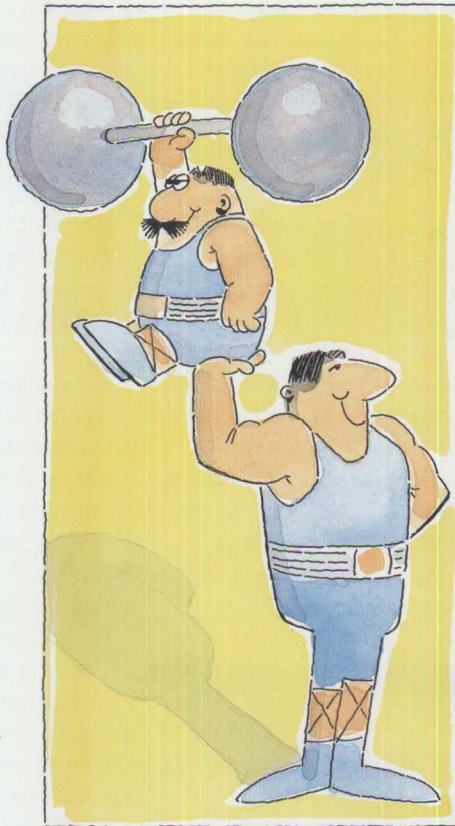
Power management of the L40 SX is the heart of what makes it a laptop and is pervasive in its system design. Special power-saving designs used throughout the machine determine when to reduce or eliminate power to various areas of the system. No other PS/2 model does this.

The L40 SX has many power management features. Some are automatic, and users are unaware that power is being adjusted within the system. Others are controlled only by some specific action.

Sleep Mode: Sleep mode is the heart of the core logic's power management while applications are running. Normally, the system processor executes an application at 20 MHz. However, with complementary metal oxide semiconductor (CMOS) technology, power consumption is proportional to system frequency. Therefore, every opportunity is taken to reduce the system frequency transparently. Sleep mode is used as long as the economy switch is set to Automatic (A).

There are two types of sleep modes — high-performance and long battery life. The set features utility determines which type of sleep mode will occur. Although the default is "high performance," the mode can be adjusted depending on application requirements.

For the long battery life sleep mode, the core logic can reduce system speed to 5 MHz. If I/O activity does not occur after some predetermined time, the system processor speed will be lowered. The amount of time that the device must be idle is different for various types of devices. The length of time for the system to remain at 20 MHz ranges from one



second on a HDD interrupt to only 1 ms on a video write. The system is tuned so database applications using the hard disk extensively, and text editors using the keyboard and mouse, will not see any apparent performance difference as the system processor speed changes.

The second phase occurs for either high-performance or long battery life sleep modes. When the application is idle, it may call one of several BIOS or OS/2 services. Special power management BIOS code waits for these calls. When one occurs, the BIOS reduces the support logic's clock to 0 Hz and turns off system processor power. An OS/2 device driver performs an equivalent function.

Turning off power to the system processor is no small matter. All vital system processor registers must be saved in an extended BIOS data area. Any interrupt that occurs reverses the process, and the system processor continues executing as if nothing happened. This process takes less than 90 microseconds and prevents any loss from high-speed data transfers.

A BIOS function at Interrupt 15h, AH=41h, is available for applications that can predict when they will be idle. If used by an application, the BIOS function helps extend battery life. When called, the L40 SX immediately reduces the support logic's speed to 0 Hz, saves the system state, and powers off the system processor. It returns from this call when any interrupts occur (timer ticks, keys pressed, and so forth). To the application, it appears like a No Operation (NOP). The BIOS call can be used on any personal computer or PS/2 and is ignored if power management is not implemented. If used, the extra power savings can extend any laptop's battery life.

A unique power management program is active under OS/2. This program is run as the lowest priority task. When this program is called, a check is made to see whether the time that the program is called is less than the minimum time for an OS/2 time slice. If it is, then a system sleep mode is requested. This way, OS/2 users can extend battery life like DOS users.

I/O Sleep: I/O devices also use a significant portion of the system power. Both automatic timeouts and user controls allow I/O devices to be powered off.

Like the core logic timeout, the I/O devices automatically shut off if not used within a predetermined time. The timeouts for the LCD, hard disk motor, and the entire system can be adjusted from one minute up to 20 minutes with the set features utility. The maximum timeout is device-dependent. Timeouts can be disabled by setting them to 0 minutes. In addition, the floppy disk controller and drive electronics turn off if not active within two seconds.

The LCD and hard disk motor timeouts are created with timers built into the core logic and integrated drive electronics, respectively. The LCD timer is reset with any keyboard or mouse interrupt, and the hard disk timer is reset with any disk activity. The system timeout, however, is created by the power management BIOS, which counts "watchdog" interrupts from the SMCU. This IRQ11 interrupt is generated by the SMCU every 100 ms whenever the system timeout is enabled.

User controls are provided to turn off the built-in serial and parallel ports as well as the optional data/fax modem or second serial adapter. When the serial ports are powered off, the associated RS-232D drivers are also powered off. Because the serial and parallel ports use the same core logic chip, the chip remains active as long as any one of the devices is on. However, when all serial and parallel ports are off, the chip is put into a special low-power standby state with all clocks turned off.

Finally, the VGA contains a digital-to-analog converter (DAC), which is required only when an external CRT is attached. When the CRT is not connected, the DAC is automatically powered off, which further reduces system current.

Suspend and Resume: Suspend and resume adds convenience to the L40 SX. With it, you can immediately turn off the system without losing data and restarting it quickly without rebooting.

The suspend operation stops the system processor, turns off all possible devices, and freezes the system state in DRAM. The DRAM is put into a slow (64 ms), low-power refresh while suspended. The DRAM and its support logic are the only components where power remains on. When the system resumes, the entire system state is restored, and the application continues as if it never stopped.

Suspend and resume adds convenience to the L40 SX.

Suspend can be initiated as a result one of the following:

- Closing the lid
- Reaching system timeout
- The SMCU detecting a low battery condition
- The SMCU detecting that the temperature limit has been exceeded

Resume can be initiated from opening the lid, an interrupt from the Real Time Clock (RTC), or by ringing the data/fax modem.

All events that create a suspend are directed through the IRQ11 interrupt. Before entering suspend, power management BIOS ensures that all I/O devices are in a stable state,

which is done by checking for pending interrupt or DMA activity. If any device has not finished being serviced, BIOS requests that the SMCU retrigger the suspend request a short time later. This retriggering is normally not visible, but continues as long as I/O activities are in progress. The BIOS suspend request guarantees that data transfers have stopped and no data will be lost.

For the system processor, suspend is similar to the sleep mode. The system processor state is saved in an extended BIOS data area, then powered off, and the clock for the system processor support logic is set to 0 Hz. The remainder of the support logic is also shut down upon disabling all clocks. Only the 32 KHz clock for the Real Time Clock (RTC) remains on. In this condition, the entire quiescent current for all system board logic totals just a few milliamperes. The 32 KHz clock is routed to the memory controller to create a slow (64-ms) refresh to conserve power. Standard high-powered DRAMs cannot support this slow refresh; therefore, the L40 SX SIMMs have a special key that prevents standard DRAMs from being used accidentally.

For the VGA, the slow 32-KHz refresh is also routed to the VGA controller for the VRAM. In addition, the contents of the palette registers within the RAMDAC are saved within the extended BIOS data area before its power is turned off. Power is removed because the DAC portion contains analog circuits that draw current even when not clocked. Simply removing the clocks to that portion would not save sufficient power for suspend. All other VGA clocks are disabled so the other CMOS logic on the chip sits idle at its quiescent current.

For other I/O devices, suspend is mostly a matter of powering them off. Special care must be given, however, to resuming the device. For example, in the floppy disk controller, all parameters are held in a static powered-on chip during suspend. Therefore, none of these parameters need be saved. For the data/fax modem, however, the modem's parameters set up by the user cannot be predicted. Therefore, they must be saved before all modem logic is powered off. A special mechanism is built into the data/fax modem for the sole purpose of dumping the modem's state prior to the suspend mode and restoring it for the resume mode.

Once all of the system state is safely saved to memory, a special "shut-down code" is saved in CMOS, a tone sounds, the suspend icon is illuminated, and the SMCU turns itself off by halting and stopping its own oscillator. This is done to save even more power. During suspend mode, attaching an AC adapter causes only the SMCU to wake up. This happens because the SMCU also supports the algorithm used to properly charge the NiCd battery.

To resume, a system reset occurs. POST first checks the previous shut-down code in CMOS to determine what caused the reset. Seeing that the last action was a suspend, it begins restoring all of the system state saved to the extended BIOS data area. The last thing restored is all of the system processor registers used by the application. POST then returns from the interrupt that originally created the suspend, and the application continues exactly where it left off.

Under OS/2, a unique power management device driver is active. This driver supports the suspend and

SMCU hardware interrupts, and guarantees that additional 80386 protected-mode registers are safely saved. With this driver, suspend and resume operate in OS/2 just like they do in DOS.

Environmental Factors

Portable computers are exposed to a rougher environment than other computers because they can be moved easily and used anywhere. Therefore, the L40 SX was designed to survive rough treatment as well as extreme temperature, humidity, and pressure swings. For example, taking the computer from the back seat of a car during the summer and going inside an air-conditioned room entails great temperature change, which can cause severe damage to many of the subsystems. The machine should not be used outside of its specified operating range of +5° to +35° Celsius (41° to 95° Fahrenheit).

Moving a machine from a hot to cold environment or using it where condensation can form could severely damage many of its parts. As a safeguard, the L40 SX was designed to detect the internal temperature of the machine and the humidity. This was accomplished by adding a thermistor and dew sensor that are continuously monitored on the SMCU. Data is interpreted differently by the SMCU, as shown in two sample operating conditions.

Condition # 1:

The system is off and is turned on.

When temperature or humidity limits, or both, have been exceeded, the machine immediately goes into a halt state and the system will not operate. In this case, the temperature or dewpoint icon, or both, turns on. The system stays in this state until

the internal machine temperature or humidity, or both, are within the machine's operating limits. This can be likened to a Power-On Self Test (POST) error, except that in this case none of the subsystems is turned on and the logic is only partially on.

Condition # 2:

The system is being used, and the temperature or humidity limit, or both, is reached.

If the machine's internal temperature drops below 5° C, the temperature icon turns on and the machine goes into a suspended state. The icon will not turn off and allow the machine to exit its suspended state until the temperature rises above 6° C. Conversely, when the internal temperature exceeds 46° C, the temperature icon turns on, and the main battery stops charging. The internal temperature must drop below 44° C before the icon is turned off. When the internal temperature reaches 52° C, the machine goes into suspended mode and will not come out of it until the temperature reaches 44° C.

If the inside of the machine exceeds 94 to 97 percent relative humidity, the dewpoint icon warns the user of the hazard. But unlike the temperature sensor, this condition will not put the machine into the suspend mode. The icon stays on until the humidity inside the system decreases to between 80 and 93 percent relative humidity.

The system was also designed to minimize the number of mechanical parts. When compared to other laptops available today, the L40 SX has significantly fewer parts. This allows the machine to withstand more operational shock and vibration, and damage from most accidents.

Battery Controls

The SMCU monitors the NiCd battery to create a battery fuel gauge on the icon panel. This function is a complex process, which relies on dynamically monitoring the battery voltage, current, and temperature as the system operates. The voltage discharge profile for a NiCd battery falls steeply during the first few moments of use after a full charge, is rather stable during the majority of its life, and becomes steep again just before it dies. The voltage also changes dramatically depending on system activity and the current being drawn. Also, the measurements must be filtered algorithmically to remove inaccuracies due to transients caused by changing loads. Finally, hysteresis must be applied to prevent false readings. The final results are shown on the icon display. A low-battery condition may trigger warning tones from the SMCU or a suspend if the warning is ignored.

The SMCU also controls the NiCd battery during the charging process. Charging characteristics of the battery are built into the SMCU. It uses the difference between the ambient and the battery temperatures to calculate the rise in temperature within the battery, which resulted from charging. The SMCU also measures the voltage and overall charging time. The result is an optimum charge that helps prevent memory effects from occurring. The SMCU also indicates from the icon display the status of the charge.

Usability and Ergonomic Considerations

Like many laptops, the L40 SX is designed with usability in mind. The keyboard layout, industrial design, and power management interface features were decided upon after extensive testing.

In addition to the traditional hard disk and floppy disk access, the L40 SX has many special features. An LCD icon status panel displays real-time information, including a modem/fax ring indicator, environmental state, speaker "on" for the hearing-impaired, battery fuel gauge, and suspend/resume state. All system connectors are placed in the rear of the machine behind latched doors that hide the connectors when not in use.

Besides the L40 SX's small size and low weight, the machine comes in a small, lightweight leather case contoured for easy carrying, and has a removable shoulder strap.

Because this machine can be used in a DC environment, its power management system is designed to extend battery life and to avoid accidental power loss. The battery fuel gauge icon has a three-stage indicator that provides audible and visible feedback when the battery is low. Once activated, the system can be used until the battery dies. Once the battery dies, the system drops into suspend mode, saving the system environment and maintaining power to the system main memory. The user needs to remove only a main battery and install a new one. The system resumes exactly where the application left off.

Summary

The IBM PS/2 L40 SX gives users the flexibility of replacing their desktop machines with a truly portable alternative. Users can now have desktop performance inside their briefcases and set up offices anywhere. The function-packed L40 SX has the usability and power to support computing requirements for many years to come.

ABOUT THE AUTHORS

Leo L. Suarez is a senior engineer and program manager for Portable System Development at IBM's Entry Systems Division Laboratory. He joined IBM in 1978, working as a component test engineer on the Series/1 and workstations. He has since held positions ranging from division headquarters technology staff to numerous management assignments in both development and manufacturing. Leo holds B.S. and M.S. degrees in electrical engineering from the University of Miami.

Frank J. Canova is a senior engineer at IBM's Entry Systems Division Laboratory, responsible for PS/2 L40 SX system design and architecture. His experience includes system design of the PS/2 P70, ESD storage subsystem architecture, IBM PC Convertible architecture, and Series/1 hard disk cache controllers. He was responsible for Asian Common User Access (CUA) design at the IBM Yamato Laboratory in Japan. Frank has a B.S. in electrical engineering from Florida Institute of Technology.

Neil A. Katz is a senior engineer and program manager at IBM's Entry Systems Division Laboratory, responsible for PS/2 L40 SX system design and technical strategy. He has held a variety of engineering assignments in Series/1 development and management positions in Communications Systems both within the U.S. and in Japan. Neil earned a B.S. in electrical engineering from the University of Florida, and an M.B.A. from Nova University.

OS/2 2.0 Considerations

IBM is planning to release a new version of OS/2 in the fourth-quarter 1991. In order for our customers to plan their future installations, this article discusses features, positioning, application migration, and future directions for OS/2 2.0.

IBM's systems software strategy has been, and will continue to be, one of providing productive operating environments that meet our customers' needs today and in the future. OS/2 is an integral part of that strategy, and IBM is firmly committed to enhancing OS/2 to meet our customers' requirements and exceed their quality expectations.

To support personal systems, DOS, DOS/Windows, and OS/2 are available. These operating systems should be selected based on individual workstation environments and usage requirements.

DOS will remain a viable operating system for many years for entry-level usage. Windows 3.X has enhanced DOS with a graphical interface and the ability to run multiple DOS applications on 386- and 486-based systems.

OS/2 1.3 is ideally suited for 286, 386, and 486 users with intermediate and advanced-level usage requirements. Its memory requirement of only 2 MB makes it ideal as a client.

OS/2 2.0 will be a robust operating system for 386 and 486 users. It will run multiple DOS, Windows, and 16-bit and 32-bit OS/2 applications concurrently. It will include a new object-based graphical OS/2 2.0 shell to exploit 32-bit hardware technology. In other words, IBM is building a technologically superior product in OS/2 2.0 that will be the "integration platform" on which our customers will execute their current software while developing 32-bit applications. IBM intends to make OS/2 2.0 available in the fourth-quarter 1991.

Features Comparison

OS/2 2.0 and Windows 3.0 appear similar, and both:

- Have a graphical user interface (GUI)
- Enable applications to take advantage of larger memory in protected mode
- Provide concurrent execution of multiple applications
- Support execution of multiple DOS sessions

There are, however, significant differences. In order to make the infor-

mation as current as possible, the expected capabilities of Windows 3.1, as made public by Microsoft, will be used in this comparison.

Usability: OS/2 2.0 and Windows 3.1 will be very usable products, and both provide:

- Graphical installation that takes maximum advantage of the benefits of the graphical user interface (GUI)
- An object-based interface with a drag or drop environment that is consistent across the system
- Support for accessing LAN resources, built directly into the shell
- Intelligent font technology
- Support for a large number of printers
- An interactive tutorial
- Simple productivity applications, games, and utilities (users can learn the system and get to work right away)

OS/2 2.0 will include a superset of those provided with Windows 3.1. However, there are differences

This paper discusses future products or future releases of products currently commercially available. The discussion regarding Windows™ is based upon information that the Microsoft® Corporation has made publicly available and is subject to change. The descriptions and discussion of IBM's future products, performance, functions, and availability are based upon IBM's current intent and are also subject to change.

Unless specifically qualified as IBM DOS, any use of DOS in this article refers generically to both IBM DOS and MS-DOS®.

between the two. Some usability features unique to OS/2 2.0 are:

- An enhanced user interface, known as the OS/2 2.0 new shell, built on Presentation Manager™ (PM) and built around the notion of objects
- Extensive on-line help and context-sensitive help for the system
- Support for long file names in the High Performance File System
- File system support for object-oriented features by means of extended attributes (EAs)

Of all these differences, the most significant for use on the system is the OS/2 new shell. It represents a significant advancement in the GUI desktop environment.

The current OS/2 1.3 shell consists of five logical units:

- Desktop Manager
- File Manager
- Print Manager
- Control Panel
- Task List

The OS/2 2.0 shell has been redesigned to give users a single interface to manage multiple types of objects, including devices (printers and drives), files, and programs. Each defined printer or attached drive is a separate icon. These objects can be arranged on the desktop or in specific shell windows. Users can interact with the objects using a well-defined drag and drop environment and manipulate files without concern for the file directory hierarchy. However, if the user wishes, multiple directory trees can be accessed simultaneously. Views, selection techniques, and actions are consistent throughout the shell.

The desktop contents and shell are saved at shutdown and restored at start-up, preserving the continuity of the work. Applications can be integrated with the shell. An application object can be the source of a drag to the shell, and an application window can be the target for a drag from the shell. When the user acts on an application object from the shell, the associated application defines how the object will respond via a dynamic link function. The shell provides default-handling for most actions on file objects.

The OS/2 2.0 shell has been redesigned to give users a single interface to manage multiple types of objects.

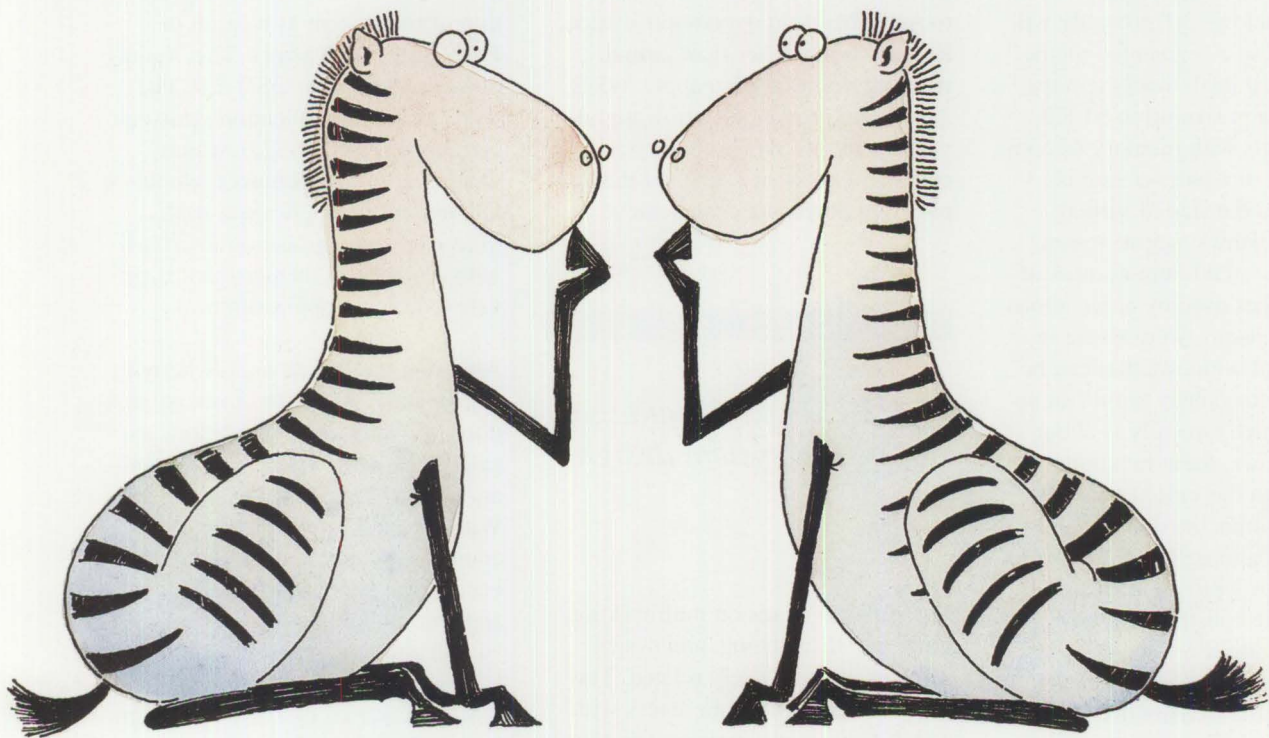
Software Technology: Figure 1 shows how some of the underlying technologies in OS/2 2.0 and Windows 3.1 affect use of the system. Windows is configured in 386 enhanced operation mode, which maximizes its capabilities, and is the default on a 386 or 486 system with at least 2 MB of memory.

Memory technology is fundamental to understanding how large an application can be, how much data it can handle, and how many applications can be harnessed simultaneously to obtain the needed solution. Memory management in OS/2 and Windows leads to some complex technical considerations. IBM believes that OS/2 2.0 will provide a much larger and

more flexible upper limit than Windows 3.1.

Systems implementing virtual paged memory, as both OS/2 2.0 and Windows 3.01 will do, have two types of memory limits. These limits are physical memory installed and virtual memory for running applications that affect how many applications can be loaded and run concurrently. Even though applications see only virtual memory, physical memory remains important. This is because performance degrades as the amount of virtual memory used begins to exceed the amount of physical memory installed. Because virtual memory is paged into physical memory by the operating system when needed, and if there is not enough physical memory, the system spends more time shuffling memory pages and less time running applications. The optimum ratio of virtual memory to physical memory is highly dependent upon the way applications are written and used.

While Windows 3.0 supports a maximum 16 MB of physical memory, both OS/2 2.0 and Windows 3.1 are intended to support greater than 16 MB, up to the physical limits supported by the hardware. This leaves plenty of room for growth, because Windows 3.1 and OS/2 2.0 are expected to need only 3 to 6 MB of physical memory. For Windows, the maximum amount of virtual memory shared across all applications is limited to four times the amount of physical memory installed, subject to available disk space. In OS/2 2.0, the limit on virtual memory is the amount of available disk space. Each application running under OS/2 2.0 has a 512 MB limit with up to 240 applications running and no limit on the total virtual memory used.



	Windows 3.1	OS/2 2.0
Physical memory limit	> 16 MB	> 16 MB
Virtual memory limit	4 X physical	512 MB (disk space)
Memory model	Segmented (64 KB)	Flat memory objects
Multitasking – DOS applications	Time slicing	Pre-emptive time slicing
Multitasking – Windows/PM applications	Cooperative	Pre-emptive time slicing
Priority	Static (set by user)	Dynamic
Dispatchability	Process	Thread
System services	Serial	Parallel
Protection between applications	Unprotected	Protected
Kernel protection – DOS applications	Unprotected	Protected
Kernel protection – Windows/PM applications	Protected	Protected
File system	FAT	Enhanced FAT HPFS installable
Reliability/availability/service support	None included in 3.0	Stand-alone dump error Logging trace utilities

Figure 1. Windows 3.1 and OS/2 2.0 Technology Comparison

Just as important as the maximum virtual memory is the model for managing it. Windows 3.1 probably will continue to use a segmented memory model that deals with memory as pieces in any size up to 64 KB. OS/2 2.0 deals with memory as complete objects of whatever size is needed, up to the size of virtual memory. Windows' segment size constrains the maximum number of many resources used by applications or Windows itself. An example is the number of windows that can be opened simultaneously within an application, which typically is about 10. In Windows, these maximums are equal with the virtual memory size, which limits the number and size of applications. The flat 32-bit memory objects provided by OS/2 2.0 avoids the constraints imposed by segmentation. For other reasons, resource maximums still occur in OS/2 2.0. These maximums are larger and not the result of a fixed constraint imposed by the memory technology.

The memory technology in Windows 3.1 should continue to provide relief from many of the constraints imposed by DOS. Windows 3.1 will retain memory technology limits that are less than the 386 hardware capability. OS/2 2.0 memory technology takes advantage of the hardware. For example, consider a system with 4 MB of physical memory. Under Windows 3.1, there is an absolute limit at 16 MB of virtual memory. Applications needing even one more byte of memory will not run.

With OS/2 2.0, there is no absolute limit at 16 MB, although with only 4 MB of physical memory, some performance limit would be reached before attaining the virtual memory limit of 512 MB. However, in OS/2 2.0, this performance limit is soft, in the sense that performance slows

down as more virtual memory is needed, yet the application continues to run. This is an oversimplification, because limits other than virtual memory size may also apply. OS/2 2.0 should provide a significant advantage in removing memory as a constraint on what users and the application developers can achieve.

OS/2 2.0 has advantages in its file system design.

The series of items on multitasking, priorities, dispatching, and system services are all closely related. The significance is the smoothness with which the system operates, and the responsiveness that applications can exhibit. On lightly loaded systems, both approaches should work well. The OS/2 2.0 technology should better support heavy workloads. This is especially important with high-speed concurrent communications programs. Multitasking allows users to continue with something new while the system is completing the previous task. A simple, instructive experiment would be to spool a large file to the printer in Windows 3.X from the file manager or a Windows application, then switch to another application and continue working. Compare this with the equivalent operation on OS/2 2.0, and judge the relative smoothness and responsiveness.

OS/2 2.0 is intended to be a very stable platform to use. OS/2 2.0 utilizes the processor's protection features to protect applications from each other and the operating system kernel

from the applications. One limitation of OS/2 1.3 was the lack of protection of the system as a result of DOS application errors. This limitation is removed on OS/2 2.0. The worst possible application error that could occur in OS/2 2.0, which would be in rare instances, would be to hang up the application itself. Moreover, no application interferes with any data in memory that is private to another application.

Although the Windows 3.X kernel enjoys some protection from applications, all Windows applications execute in the same address space and are not protected from each other. Windows 3.X applications risk encountering data integrity problems because an addressing error in one application could modify data private in another and be undetected. Likewise, key portions of the Windows 3.X kernel can be overwritten by a DOS application, resulting in the unknown application error (UAE) message and a request to reboot the system. The importance of having good protection is clear to anyone who has lost work from multiple applications because one of them managed to hang the system.

OS/2 2.0 has advantages in its file system design. First, file systems are installable, simplifying support of new storage media requiring specialized functions. An example is the installable file system for CD-ROM. Second, OS/2 2.0 provides the High Performance File System. Its major features are:

- Advanced strategies for laying files out on disk to minimize fragmentation and maximize disk performance
- Exploitation of SCSI hardware performance features
- Sophisticated caching for reading and writing

- Support of very large disks
- Support for long file names

An enhanced FAT file system is supported for floppy diskettes and for compatibility with existing hard disks already using FAT. Numerous performance improvements have been made relative to OS/2 1.3 and DOS while maintaining compatibility.

Standard operating system support for reliability, availability, and service (RAS) is another attribute of OS/2 2.0. There are utilities to isolate and log system software problems. The benefit is that software service is integrated into OS/2 2.0. Windows 3.0 did not include any RAS-like features. As an environment, OS/2 2.0 should provide many advantages, both in usability and in the robust technology underlying it.

DOS Application Support: Because of the large base of available DOS applications, it is necessary to focus on a detailed comparison of DOS application support. Figure 2 shows a breakdown of the key environments, including DOS and OS/2 1.3. Windows 3.0 was also included so several key measurements can be made.

The comparison applies to operation on a 386 or both. Windows 3.0 is normally not run in real or standard mode on this class of machine, especially if the emphasis is on DOS applications. Those environments are included for reference, because real mode is required to run Windows applications that have not been upgraded to support Windows 3.0. Standard mode may give better performance in systems with limited memory.

The first group of entries relates to the raw memory space seen by the DOS application. Conventional memory remains in an application after subtracting the memory "overhead" used by the system code. The values given are typical for a default installation. IBM DOS 5.0 appears to have more conventional memory because the default DOS installation does not include expanded memory (EMS) or mouse support.

Standard operating system support for reliability, availability, and service is another attribute of OS/2 2.0.

In Windows 386 enhanced mode, the default installation provides EMS support. The OS/2 2.0 default installation provides both EMS and mouse support. Including support for EMS (at 8 KB) and mouse (at 14 KB) reduces the available conventional memory in DOS to 601 KB. In addition, any resident software or device drivers for functions such as LAN or host connectivity are subtracted from the conventional memory available to DOS applications under DOS or Windows. However, they do not affect the memory available in OS/2 2.0.

This is illustrated by an example showing memory remaining with a LAN requester installed. EMS support is through emulation of the EMS 4.0 specification out of extended memory. Note that Windows 3.0 has a fixed pool of memory allocated among all applications, while

OS/2 2.0 has a separate, larger limit for each individual application. The OS/2 2.0 limit on extended memory available to a DOS application is based on using the extended memory specification (XMS) interface. DOS applications using the DOS protected-mode interface (DPMI) specification can access up to 512 MB of extended memory.

The next group of entries covers memory management characteristics. Physical RAM describes the physical memory locations available for loading the DOS application. Where only memory below 1 MB is available, only one DOS application can be in memory at a time, and the other DOS sessions are swapped out to disk.

"Memory overcommit" is the capability to run applications needing more memory than is physically available on the machine. The notation "none/switch" means that no individual DOS application can overcommit memory, but the real mode portion can be moved to disk to make room for another DOS application. However, extended or EMS memory allocated by the application is not switched to disk.

OS/2 1.3 can swap the DOS application to disk when running protected-mode applications. Windows 3.0 386 enhanced mode can overcommit up to four times the physical memory on the machine. OS/2 2.0 is limited only by the amount of available disk space.

Although the default is to swap through the file system, Windows 386 enhanced mode allows a swap space to be preallocated, which leads to improved performance by avoiding the DOS file system. Because all of this disk space is pre-allocated, none can be shared

	IBM DOS 5.0	OS/2 1.3	Windows 3.0 on IBM DOS 5.0			OS/2 2.0
			Real	Standard	Enhanced	
Conventional memory – with EMS and mouse	623 KB 601 KB	529 KB	558 KB	571 KB	569 KB	633 KB
Memory with LAN attachment	543 KB	486 KB	478 KB	491 KB	489 KB	633 KB
DOS Extended Memory (XMS)	16 MB	none	16 MB (total)	16 MB (total)	16 MB (total)	16 MB (per application)
DOS EMS 4.0 memory	16 MB	none	16 MB (total)	none	16 MB (total)	32 MB (per application)
Physical RAM for DOS	0 - 1 MB	0 - 640 KB	0 - 1 MB	0 - 1 MB	Any/Paged	Any/Paged
Memory overcommit	None/Switch	None/Swap	None/Switch	None/Switch	4 x RAM	Avail Disk
Swap file	File System	File System	File System	File System	Physical or File System	File System
Number of DOS applications	16	1	16	16	16	>32
Background execution	No	No	No	No	Yes	Yes
Invocation	Shell/Cmd	Icon	Icon	Icon	Icon	Icon
Windowed	No	No	No	No	Yes	Yes
Cut and paste	No	No	Yes	Yes	Yes	Yes
Print spooling	Yes	Yes	No	No	No	Yes
Installable file system	No	Yes	No	No	No	Yes
Direct hardware access	Yes	Yes	Yes	Yes	Yes	Yes
Timing-dependent applications.	Foreground	Foreground	Foreground	Foreground	Exclusive- Mode Option	Foreground/ Background
DOS Protected-Mode Interface	No	No	No	No	Yes	Yes
Virtual Control Program Interface and DOS Extenders	Yes	No	No	No	No	No
Continues after Serious Application Errors	Rarely	Rarely	Rarely	Rarely	Sometimes	Usually

Figure 2. Comparison of DOS Environments

(Typical values for an IBM Model 8580-071)

dynamically. OS/2 2.0 implements access to the swap space via the file system for both FAT and the High Performance File System. The OS/2 2.0 implementation should provide the benefit of a dynamically sized swap file combined with good performance.

Continuing down the table, a maximum of 240 DOS applications runs under OS/2 2.0. This number is equal to the maximum number of applications. In practice, few people will reach the suggested limit of 32. OS/2 2.0 provides windowing of DOS applications on the PM desktop in all text and VGA graphics modes. Windows 3.0 386 enhanced mode, however, supports windowing of DOS applications for text and CGA graphics.

Although Windows includes a print spooler, its services are not available to DOS applications, and printing concurrently from DOS applications is not supported. Windows permits only one DOS application to print and requires that other DOS applications are suspended if they attempt to print concurrently.

Using a DOS print spooler (loaded prior to Windows) is not a viable solution, because printing concurrently from multiple DOS sessions causes output to become intermixed on the same page. Windows warns of a device conflict while using the printer and offers a choice on how to proceed. But regardless of the choice, the same incorrect output appears. OS/2 2.0 provides correct spooling of printer output from concurrent DOS applications.

Under DOS, many specialized devices are supported directly from the application without a device driver. The ability to run these applications

is under the heading "direct H/W access."

Similarly, some DOS applications are extremely timing-sensitive, mostly in the area of communications applications using high data rates. These are supported in all environments when the application is in the foreground. However, in DOS, OS/2 1.3, Windows real mode, and Windows standard mode, the application is suspended while in the background and cannot satisfy any timing constraints.

OS/2 2.0 implements access to the swap space via the file system for both FAT and the High Performance File System.

In 386 enhanced mode, Windows 3.0 provides the option of setting exclusive mode, so a timing-sensitive application can be run in the foreground without interference due to time slicing. However, all other applications are suspended and do not run while this DOS application is running in exclusive mode. In some cases, the need for exclusive mode can be avoided by manually adjusting the relative application priorities. The ability to dynamically manage priorities permits OS/2 2.0 to multi-task timing of critical applications both in the foreground and background.

Some DOS applications include a technology known as DOS extenders that allow the application to execute out of extended memory. DOS ex-

tenders based on the DPAPI specification are supported under all environments except OS/2 1.3.

Other extenders, including those based on the Virtual Control Program Interface (VCPI) specification, are not supported outside of DOS itself. These types of extenders are inconsistent with the mechanisms used in OS/2 2.0 to protect the system or other applications from application errors.

Most of the environments rarely continue after a serious application error, because a failure that hangs a DOS application usually also hangs the entire system. In 386 enhanced mode, Windows 3.0 can sometimes continue after a serious application error, but will fail when the application has overwritten portions of DOS or Windows that are in the DOS address space. Windows usually advises the user to shut down and restart Windows, because the errant DOS application may have left the system in an unstable state. Because of the protection mechanisms in OS/2 2.0, a DOS application error should not affect anything in the system other than its own session, allowing OS/2 to continue after application errors occur.

Therefore, when it comes to running DOS applications, OS/2 2.0 should clearly be superior to both the Windows 3.0 and DOS environments.

Market Positioning

To best understand the marketplace, consider the wide variety of workstation environments, usage levels, and existing configuration needs. Then, look at IBM's suggested positioning of the operating systems in relation to those needs.

Environment: Although actual use of personal computers varies widely,

<p><u>Stand-alone Workstation</u></p> <ul style="list-style-type: none"> • No LAN attachment • May have asynchronous communications • Personal productivity and stand-alone versions of small business departmental or special-purpose applications 	<p><u>Work Group LAN</u></p> <ul style="list-style-type: none"> • LAN-based resource sharing • Multiple work group LANs (via bridges) • Non-SNA host communications • Multi-vendor hardware and software • Local and distributed personal productivity, mail, small business, departmental, and special-purpose applications • Server-based systems management
<p><u>Enterprise Workstation</u></p> <ul style="list-style-type: none"> • No LAN • Controller-based connection to host (mid-range, mainframe) • Local personal productivity and host applications/mail 	<p><u>Enterprise LAN</u></p> <ul style="list-style-type: none"> • Large LAN networks • High-volume, high-performance networks • SNA host communications • Multi-vendor hardware and software • Work group applications, distributed host applications/mail • Host-controlled system and network management

Figure 3. Workstation Environments

it is helpful to classify personal computer use into the four workstation environments shown in Figure 3.

There are various levels of usage for personal computers, which can be described with the following usage-level characteristics:

Entry – uses personal productivity applications one at a time. May have a single asynchronous, LAN, or SNA communications session. System typically has 2 MB or less of memory.

Intermediate – uses personal productivity applications with expanded or extended memory. Utilizes “task switching” to switch between multiple applications. May have a single asynchronous, LAN, or SNA communications session. System typically has 2 to 4 MB of memory.

Advanced – uses multitasking, mission-critical, or complex applications as well as personal productivity applications. May use one or more simple or advanced communications sessions. System typically has 4 to 8 MB of memory.

Server – provides broad function, from print/file sharing to distributed/cooperative processing. Additional requirements include interoperability, multitasking, network management, and software distribution.

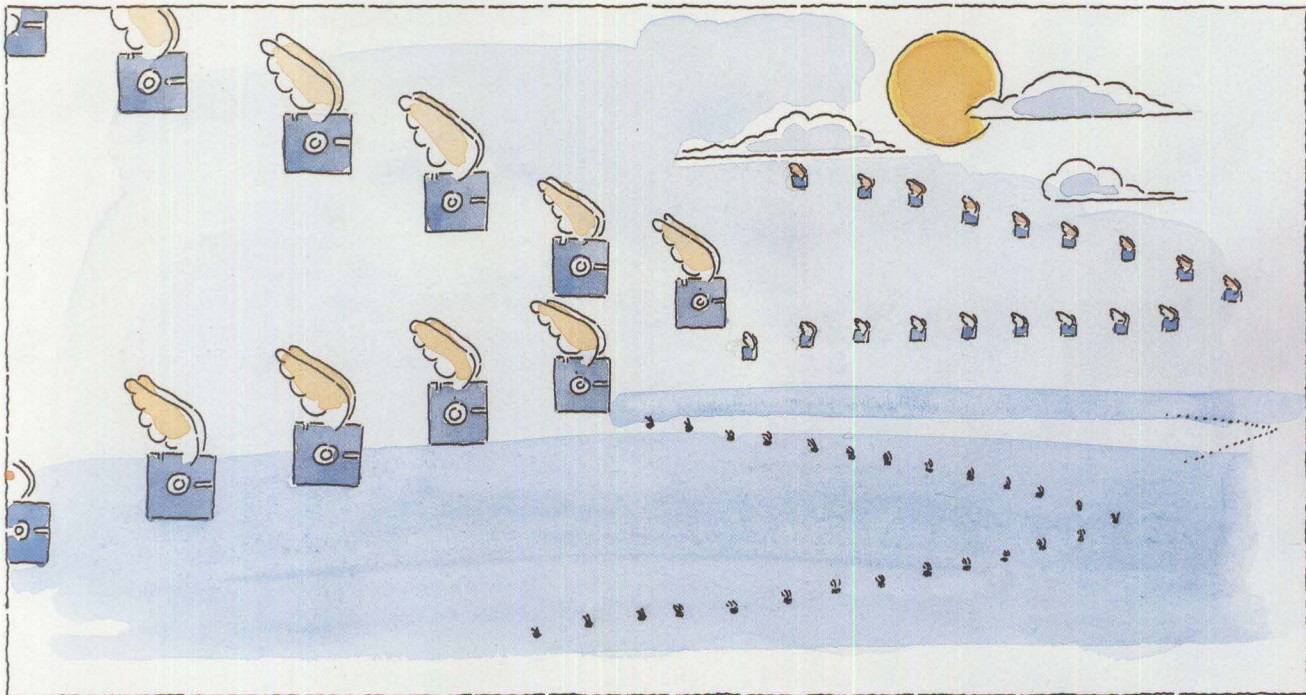
Product Positioning: The following guidelines should help you select the proper operating system:

DOS – continues as the low-end operating system of choice for entry systems, typically equipped with less than 2 MB of memory. DOS should

be selected for entry usage in all four workstation environments.

DOS/Windows 3.0 – provides an excellent extension to DOS for systems containing 2 to 6 MB of memory. This graphical Windows environment will continue to satisfy entry-level needs in the foreseeable future. Given the significant investment in 8088, 8086, and 286 technology, DOS/Windows can satisfy the investment return in the short term until the move is made to 32-bit technology. DOS/Windows should be selected for entry-level usage in all four workstation environments and for intermediate usage in small businesses in stand-alone workstation and workgroup LAN environments.

OS/2 1.3 (16-bit) – provides a robust operating system platform for systems containing 2 to 6 MB of memory. OS/2 1.3 should establish itself as the high-quality, preferred operating system for users of 286, 386, or higher systems. In addition, OS/2 1.3 will continue to be an SAA™ platform and will be made more flexible by packaging the Communications Manager™, Database Manager™, and LAN functions (today available only in IBM's OS/2 Extended Edition) so they will run on any Intel-based hardware platforms using IBM or non-IBM OS/2 Standard Edition. In the future, IBM intends to enhance OS/2's Communications Manager and Database Manager components by adding functions like ISDN, forward recovery, and access to DB2™ (remote unit of work gateway). OS/2 1.3 should be selected for advanced use in all four workstation environments and for intermediate use in medium/large accounts in all workstation environments. It should also be selected for servers.



OS/2 2.0 (32-bit) – will become the “Integration Platform.” Multiple DOS, Windows, and 16-bit OS/2 applications will run better on OS/2 2.0 than in their current environment for any 32-bit system (386SX, 386DX, 486) containing 3 to 8 MB of memory. Communications Manager, Database Manager, and the LAN function will also be available on this platform. Additionally, new 32-bit applications can be developed and run, leading to a fuller exploitation of the 32-bit hardware technology.

OS/2 2.0 should be selected for intermediate and advanced use in all four workstation environments. It is a candidate for any 386SX or better with at least 3 MB. It should also be used for servers.

Application Migration

Given the large number of DOS applications and the emerging numbers of Windows applications, a key con-

Additional productivity gains can be made by migrating 16-bit applications as time permits.

sideration is how well this application base runs on OS/2 2.0. This includes both continued use of existing applications and the option of using unique applications that may first appear under DOS or Windows but not OS/2. This requirement needs to be addressed from two viewpoints: user and developer.

User Perspective: The goal is for OS/2 2.0 to run DOS, Windows, and OS/2 applications better than DOS, Windows, or OS/2 1.3, in terms of

usability, integrity, and performance. Without any need to upgrade any applications, user productivity should increase after installing OS/2 2.0.

Additional productivity gains can be made by migrating 16-bit applications to 32-bit applications as time permits. IBM expects to make significant progress towards achieving its stated goals in OS/2 2.0. However, these goals may not be fully achieved until a later release of OS/2 2.0.

Let’s begin by looking at DOS applications, and then continue with Windows and OS/2 applications. Here’s how they will operate on OS/2 2.0.

DOS applications – support all DOS real-mode applications. The key DOS extensions for EMS, XMS, and DPMI services are provided by OS.2 2.0. Applications using DOS extenders to run in protected mode work only if the extender conforms to the

DPMI specification. When the extender does not conform, an upgrade to a version of the extender supporting DPMI is required.

Most applications run well without much user attention. In some cases, the advanced properties feature of the DOS environment provided by OS/2 2.0 can be used to reduce resource requirements or improve performance. In a few cases, hardware-, timing-, or DOS-dependent applications require advanced properties.

A database of recommended settings is included with OS/2 2.0 to help configure applications' advanced properties. The OS/2 new shell allows DOS applications to be started and managed directly.

DOS applications running in all text or VGA graphics modes can appear in windows on the PM desktop. Concurrent printing from multiple DOS applications is supported by the OS/2 2.0 spooler, unlike Windows 3.0, which does not provide print spooling to DOS applications.

Windows 2.1 and 3.0 applications – are supported. All necessary code is included with OS/2 2.0, including the standard Windows printer and plotter drivers. For other devices, the Windows printer or plotter drivers provided by the hardware manufacturer is required.

The goal is for Windows 3.0 applications to run together with the other applications on the PM desktop. However, this goal may not be fully achieved until a later release, and initially the Windows 3.0 applications may run on a separate Windows desktop. Windows 2.1 applications will always run on desktops of their own. Windows 3.0 requires shutting down Windows and restarting in real mode to run Windows 2.1 applica-

tions. Windows and PM applications will communicate via the clipboard protocol and via the DDE protocol.

Existing OS/2 applications – should perform better on OS/2 2.0 than on OS/2 1.3. Best of all, users can take advantage of new applications as they appear, whether they are DOS, Windows, 16-bit OS/2, or the new, more powerful OS/2 2.0 applications.

Developer Perspective: While the decisions for the user are simple and straightforward, the decisions facing the developer are considerably more complex. Should applications be developed for Windows, OS/2, or both? What about future applications? When is development of a 32-bit application an appropriate choice?

DOS applications – can be migrated to OS/2 2.0. Regardless of whether the target of the DOS port is a Windows or a PM application, the structural design differences imposed by writing to a graphical user interface requires considerable rework in the structure of a DOS application. OS/2 2.0 offers the option of porting the DOS application to a 16-bit full-screen application with minimal restructuring, just as OS/2 1.3 does. However, given the capability of running DOS applications on OS/2 2.0, the need for such a port is questionable unless the application needs to add function that would benefit directly from access to more memory or from using features provided by OS/2.

16-bit applications – will be a marketed for both Windows and OS/2 2.0. That is not as much of a problem as it might seem given the support provided the Microsoft software migration kit (SMK), often referred to as the Windows Libraries for OS/2 (WLO). As described in the *Microsoft Systems Journal*, Novem-

ber 1990, porting most Windows 3.0 applications to OS/2 is simple. Most applications designed with the port maintains a single set of readable source code. IBM is developing its own new migration package with enhancements for OS/2 2.0, both to continue improving performance and to enable missing function, like the palette manager. Experience with a recent Microsoft SMK is that the overall performance on OS/2 is only about 10 percent slower than on Windows, and performance is noticeably smoother on OS/2 when multitasking. The goal for the IBM migration package is to provide better overall performance on OS/2 2.0 than the same application running on Windows 3.0.

The OS/2 2.0 device driver kit supports porting Windows device drivers to OS/2 as easily as applications port through the IBM migration package. This makes it easy for hardware vendors supporting devices on Windows to also support these devices on OS/2. For vendors developing OS/2-unique drivers, the device driver kit also has a library of routines and a template OS/2 device driver to simplify the task.

Because OS/2 2.0 runs Windows applications directly, a natural question is, why bother releasing a PM version of the application? A PM version will be released because it provides an easy way to differentiate the application by taking advantage of the additional features of OS/2 2.0. Examples are:

- Support of long file names under the High Performance File System and improved data integrity
- Better integration into the OS/2 new shell
- Use of threads for better multitasking and responsiveness within the application

OS/2 is best-suited for applications:

- With communications and database requirements
- Requiring interprocess communications, multitasking, semaphores, multithreading, or graphics

32-bit applications – benefit from OS/2 2.0's flat memory model. This environment does not have the problems associated with 64 KB segments. The 16-bit segmented memory model forces applications to write code that depends on segmentation and manages the virtual address space. Applications with many code and data segments or requiring access to very large data objects are especially sensitive to limitations of 16-bit segmentation.

Any discussion of the benefits of a 32-bit architecture leads directly to performance. OS/2 2.0 enables exploitation of the 386 and 486. The performance improvement clearly depends on what the application does and how the application interacts with the operating system. Advanced capabilities such as full-motion video, digital sound, and speech and handwriting recognition often require complex algorithms to perform.

As demonstrated in preliminary testing of performance-sensitive areas of OS/2 2.0, there should be a significant benefit by using native 386 instructions with 32-bit operands. Low-level functions such as string manipulation, 32-bit integer math, data movement, and pointer operations should be approximately twice as fast as 16-bit equivalent instructions. Applications written in C should achieve 5 to 15 percent improvement with minor changes and recompilation. Simple memory management changes should equate to an additional five percent of performance. Applications fully imple-

16-bit OS/2	32-bit OS/2
Multitasking APIs	Minor modifications Some redundant APIs deleted
Signal APIs	Combined into exception management Simple to change
Exception-management APIs	Process model becomes thread model Exit processing unchanged
Pipes/queue APIs	Unchanged
Semaphore APIs	New, concise semaphore model Simple to migrate/change
Timer services	Unchanged
Memory management APIs	New APIs to manage flat memory objects Simple for most applications
File system APIs	Unchanged (except names standardized)
Session management APIs	Unchanged
Presentation Manager APIs	Minor modifications Some redundant APIs deleted
Device Driver Interface (DDI)	Unchanged

Figure 4. PM Application Migration from 16-bit to 32-bit OS/2

mented to take advantage of the flat memory and the new unique 32-bit interfaces should achieve 20 to 60 percent improvement in performance.

The flat memory allows for the elimination of loading and reloading of segment selectors for accessing fetch access register "FAR" data, FAR calls, huge arrays, and long variables. The combination of 32-bit applications using the new, optimized 32-bit APIs, like memory management and semaphores, should maximize performance. An application showing the greatest performance improvements seen to date is REXX, for example, where, in preliminary testing, the preceding produced a 60 percent improvement in performance while processing statements.

A primary motivator behind the design of the OS/2 2.0 APIs is portability. The flat, 32-bit, virtual address space is a common feature on many

platforms and is key to portability. To better compete with other high-end workstation operating platforms such as UNIX®, OS/2 2.0, including its applications, dynamic link libraries, and kernel should be retargeted to other processors, such as the RISC System/6000™. Likewise, it is desirable that applications from other platforms port easily to OS/2. The 32-bit flat memory architecture in OS/2 2.0 removes one of the key inhibitors to a successful port.

In support of portability, considerable effort has also gone into making the APIs more extensible, removing dependencies on specific processor architectures, and providing a meaningful and consistent naming convention. Figure 4 contains a summary of what can be expected when porting an OS/2 application from 16-bit to 32-bit.

Another case is creating 32-bit versions of Windows applications. This

16-bit Windows	32-bit Windows	32-bit OS/2
User APIs	All 16-bit values become 32 bit, including message parameter and handles. Care is needed with window words. Input model changes Message reordering	Similar changes No change
GDI APIs	32-bit coordinate system Many graphics applications affected 60-plus API calls changed Some API calls removed	16-bit coordinate system available (32-bit optional) GDI to GPI change similar
Multitasking APIs	All new APIs	Similar changes
Pipes APIs	DOS calls turned into Windows APIs	Similar changes into OS/2-based APIs
Timer services	??	Use OS/2-based APIs
Memory management	Change to flat memory	Similar changes
File system APIs	DOS int 21 functions turned in Windows APIs	Use OS/2-based APIs
Session management	Largely unchanged	Use OS/2 PM APIs

Figure 5. Application Migration of 16-bit Windows to 32-bit Windows or to 32-bit OS/2

decision is more complex, because Microsoft has disclosed that a 32-bit version of Windows will be re-released. A discussion of Windows 3.2 is included in the next section, which discusses future OS/2 directions. Regarding porting issues, Figure 5 shows what it means to port a 16-bit Windows application to 32-bit, either Windows or OS/2, broken down by the API groups. The information on Windows 3.2 is based on the information made publicly available by Microsoft at the time this was written.

A perception about Windows 3.2 is that it should be easy to move Windows applications to this API. A port to OS/2 is perceived as being more difficult; however, this is not necessarily so. Basically, the changes needed are similar for both Windows 3.2 and OS/2 2.0.

A lot of the complexity of moving a Windows application to exploit the

new 32-bit world deals with the new function. The rest comes with the new environment, changing parameters from 16 to 32 bits, and from segmented to flat memory. In many cases, 16-bit OS/2 PM applications can be converted to 32-bit with minimum effort. Converting 16-bit Windows applications to 32-bit requires considerable work, regardless of whether the target is Windows 3.2 or OS/2.

Within the API categories shown in Figure 5, the function provided by the Windows APIs closely matches OS/2 2.0 functionality. However, at present, the semantics of the individual API calls are different. Another consideration is that OS/2 2.0 supports mixed 16- and 32-bit applications, thereby easing migration. As of this writing, Microsoft is offering Windows 3.2 support only for 32-bit applications. Most of the function promised for Windows 3.2 will be available on OS/2 2.0 later this year.

Languages and Tools: IBM's goal is to make OS/2 2.0 the development platform of choice for both OS/2 and DOS/Windows applications. This goal will be accomplished by enabling developers to exploit the power of the newer, high-performance hardware platforms, which includes running multiple compilers for program builds and support for debugging DOS and Windows applications. The high degree of system integrity provided by OS/2 2.0 is an additional benefit. The result should be better performance and throughput with higher productivity.

IBM is working to ensure the availability of a comprehensive set of programming tools on OS/2, both from IBM and independent software vendors (ISVs). IBM intends to provide a core set of 32-bit development tools including:

- A 32-bit optimizing and ANSI conforming C compiler

- A configurable programming editor (extensible through REXX)
- A 32-bit PM interface debugger
- A project-oriented workbench with an open architecture

IBM is working with ISVs to provide the comprehensive existing 16-bit OS/2 tools in 32-bit OS/2 versions as well. The IBM tools will be provided with IBM service and support. Existing 16-bit OS/2 tools will continue to work on OS/2 2.0 for developing 16-bit applications.

No prerequisite toolkits should be needed for developing simple applications. The necessary libraries and binding files are shipped with the operating system. The base system also includes the standard IBM Procedures Language – REXX. The future direction for REXX is object technology to enhance its ease-of-use and power. A toolkit for developing more complex applications will include items like resource editors and compilers, message file creation and binding tools, the IPF compiler, system profilers, and on-line documentation.

Future OS/2 Directions

This section discusses the New Technology (NT) kernel, database, multimedia, software components, LAN solutions, and distributed environments.

NT Kernel (OS/2 3.0): Development responsibilities between IBM and Microsoft changed in September 1990. Microsoft accepted lead responsibility for the development of a new, portable kernel based on newly emerging technologies. This kernel meets Department of Defense B-level security standards and is certified at the C2 level. New file system technologies and symmetric multiprocessing are also being ex-

plored. The plan is to provide a future version of OS/2 built on top of this kernel. Microsoft has indicated it intends to support Windows and a 32-bit extension of Windows as well as PM on this kernel. Because OS/2 will also be supported on top of NT, the same capabilities will be available or can be easily provided for OS/2 applications. Most of the remaining function slated for Windows 3.2 and NT will be available in OS/2 2.0 later this year.

One of the objectives of NT is to run on non-Intel processors, notably RISC processors. While DOS, Windows (16- and 32-bit) and OS/2 (16- and 32-bit) applications will be easily supported by NT running on an Intel processor, the situation is more complex on RISC. The technology for providing DOS emulation on RISC is well-understood and is now available on the IBM RISC System/6000™. Windows and OS/2 applications that are 32-bit need to be recompiled with Assembler routines rewritten. For Windows and OS/2 16-bit applications, the memory segmentation model presents a problem. The technology for getting these applications to run well on a RISC processor, short of porting them to a 32-bit implementation, is not understood today. For this reason, 32-bit implementations appear to be a better choice today if future portability across processors is a business consideration.

Database: IBM is developing the OS/2 database to meet the industry's need for scalable high performance, robustness, reliability, fault tolerance, manageability, security, recoverability, mainframe database access, and interoperability across multiple platforms. IBM will provide full, 32-bit operation on OS/2 2.0 and support client operations under DOS, Windows, OS/2, AIX®,

and UNIX. Both ANSI SQL™ and SAA SQL compliance are goals, with support for application development in multiple high-level languages on OS/2, AIX, and UNIX.

APIs will be developed for use by applications, and front-end tools will be provided to monitor and control database operations. Work is proceeding toward supporting a distributed environment, including links to DB2 and SQL/DS™. The published distributed relational database architecture (DRDA) “language” is designed so both IBM and non-IBM developers can produce software that interfaces as either a server or a requester to DB2 initially, with subsequent support on the other IBM relational database products.

Multimedia: This is an important new area for OS/2. Development of multimedia products can be easier on OS/2 than on Windows. The experience with IBM products like Audio Visual Connection™ and M-Motion has been that Windows implementation requires significantly more effort than OS/2. This is a result of better:

- Encapsulation of system resources by OS/2 1.3; under Windows, the programmer needs to be aware of directly controlling the system resources
- Stability of OS/2 1.3, especially in the area of multitasking, which provides improved coexistence of applications with less coding effort. For example, items like “yield points” required under Windows are not necessary with OS/2 1.3
- Graphics, such as Bezier functions, on OS/2 1.3

The combination of better multitasking and the support for threads, leads to better synchronization of the

multiple data streams needed for multimedia. An example is synchronizing digital audio with analog video for a firing cannon or a talking head. OS/2 2.0 "fast threads" is an enhancement designed to improve the threads performance for multimedia applications. In Windows, synchronization is left entirely up to the application.

Multimedia applications frequently need to deal with large memory objects for bit maps, audio streams, or even streams of bit maps. These objects are much easier to manipulate in the flat 32-bit memory model provided by OS/2 2.0. Although the Windows 3.0 SDK provides the WINMEM32 DLL, which is an interface for allocating 32-bit flat memory from a Windows application, Windows remains a 16-bit segmented environment. Use of WINMEM32 leads to a mixed 16- or 32-bit application with the usual risks associated with the unprotected, global memory allocation used in Windows 3.0. As described in the SDK, all the complex issues of managing a mixed 16- or 32-bit application beyond memory allocation basics must be addressed by assembly language code provided by the developer. Moreover, the WINMEM32 DLL is not part of the retail Windows 3.0 product.

Object-Oriented Development: A significant emerging software technology is the use of objects. IBM intends to enhance OS/2 2.0 with extensions to improve support for object-oriented programming languages. Support of class libraries using an approach equivalent to dynamic link libraries (DLLs) is under consideration. Presentation services will be enhanced with more object-oriented features. In addition, there will be further object-oriented extensions to the user interface.

Patriot Partners: Looking toward the 21st century, a major challenge in the software business is to find a way to provide profitable software solutions for specialized problems. Patriot Partners was formed between IBM and Metaphor™ Computer Systems Corp. to create applications suitable for addressing this challenge. Patriot Partners' key objectives are to:

- Vastly reduce the development risk associated with selecting a specific hardware and operating system platform by enabling a single product to target multiple platforms from multiple vendors
- Enable the introduction of new software technologies, like object-oriented programming, visual programming, multimedia, and expert systems
- Provide platform-independent objects needed to build applications in an extensible development environment
- Encourage the development of applications built out of reusable components connected by well-defined protocols designed for selling the components and the application
- Provide tools known as assemblers and builders for combining components into specialized applications/solutions
- Enable better application combination by end users for solving specific business problems using visual programming and the structure provided by components and protocols

IBM intends to make the partnership technology available on OS/2 and AIX. Metaphor will provide the technology on other platforms, including other UNIX systems and the Macintosh®.

LAN Solutions: IBM will continue to strengthen its position as a provider of LAN solutions. This will be done through enhancements to current IBM products and through partnerships and cross-licensing agreements like the one recently announced with Novell® Inc. Potential LAN enhancements are in the areas of local and wide area networking, interoperability, management, and communications support from both IBM and Novell products. Examples are software distribution, selective backup/archive, transporting NetWare™ IPX/SPX packets across SNA networks, NetWare client access to Communications Manager, and supporting Communications Manager and Database Manager access to NetWare servers.

Distributed Environments: IBM recognizes the increasing importance of heterogeneous vendor support on LANs and networks, and is committed to helping its customers run their businesses with these systems. In May 1990, the Open Software Foundation™ (OSF™) announced the distributed computing environment (DCE), a selection of technologies aimed at simplifying the work for users and application developers in these complex computing environments. In endorsing DCE, IBM subsequently announced not only support for DCE on AIX, but also the intent to extend SAA to incorporate key elements of DCE. As one of the SAA systems, OS/2 will be a platform for the delivery of the key elements of DCE.

DCE consists of eight key technologies. Of these, six are of particular interest for distributed environments including OS/2, and can be summarized as follows:

Remote Procedure Call (RPC) – takes care of the communications details so that writing distributed applications approaches the simplicity of applications on a single machine. The basic notion is that procedures called by an application may actually be run on a computer elsewhere in the network.

Distributed Naming Service – provides a single naming model throughout the distributed environment. Resources such as servers, files, disks, or print queues are identified by name, independent of the physical location in the network. Full X.500 support is provided.

Time Service – provides a mechanism for synchronizing each computer in the network to a recognized time standard. Many distributed applications need a single time reference to properly determine event sequencing and duration.

Security Service – provides the network with authentication, authorization, and user account management. Authentication validates the identity of a user or service to prevent fraudulent requests. Authorization is the process of determining whether authenticated users should have access to a resource. These facilities are made available through a secure

communications capability provided by the RPC.

Threads Service – is a facility to support concurrent programming much like threads in OS/2. This will be used by other DCE components to implement their services.

Distributed File System – joins the file systems of the nodes in the network through a consistent interface that makes global file access as easy as local file access. The Distributed File System should provide users with a uniform name space, file location transparency, and high availability.

Conclusion

OS/2 2.0 will provide the features that our customers need. It will be compatible with existing DOS, Windows, and OS/2 applications and will have the performance and enhancements provided by 32-bit applications. OS/2 2.0 will provide the stable base for increasingly complex, mission-critical, and networked applications. With the OS/2 2.0 new shell model, OS/2 2.0 will provide a more productive object-oriented user interface.

Of course, DOS, Windows, and OS/2 all have a place on the workstation. In this mixed environment,

IBM will provide a simple migration path to preserve investment in existing applications as requirements on the computation platform become more complex.

Windows 3.0 provides a superset of the DOS capability, retaining the ability to support DOS applications while adding the benefits of the GUI provided by Windows applications.

With OS/2 2.0, IBM intends to provide a superset of the Windows capability, providing support for DOS and Windows applications while adding the benefits of an advanced GUI (OS/2 2.0 new shell model), greater integrity for mission-critical applications, support for complex communications requirements, and support for both 16- and 32-bit OS/2 applications.

Among these three PC operating systems, OS/2 2.0 with 32-bit applications is the only one that in 1991 will utilize the performance benefits of the 386 and 486 hardware platforms. Because of these additional benefits combined with the competitive price, OS/2 2.0 should quickly become the operating system of choice for personal systems in the 1990s.

Comparing PC-DOS, OS/2, and AIX PS/2 – Part 2

W. Craig Chambers
IBM Corporation
Dallas, Texas

The technical side of PC-DOS, OS/2, and AIX PS/2 are examined in this second installment, which compares these three operating systems.

Each of the three operating systems – PC-DOS (hereafter referred to as “DOS”), OS/2, and AIX PS/2 – have different file systems. DOS uses the file allocation table, or FAT, to allocate space on a disk. OS/2 uses this same system for compatibility with DOS, yet OS/2 Version 1.2 includes a new disk organization system called the High Performance File System (HPFS). AIX PS/2 uses a system based on control blocks called *inodes*, which are similar in concept to the OS/2 HPFS organization. Each of the file systems is described here.

Directory Structure

File systems' directory structure was briefly explained in Part 1. Figure 1 shows this structure, which has a first-level root directory followed by lower-level subdirectories.

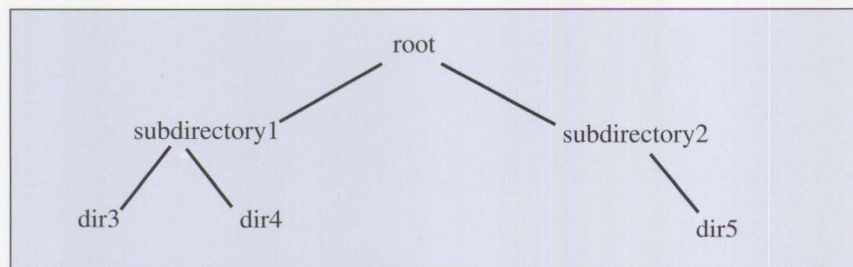


Figure 1. Directory Tree

While similar in external appearance, there are major differences between the file systems of AIX and either DOS or OS/2. In DOS and OS/2, directories can contain entries for files or subdirectories, which are simply files containing directory entries.

In FAT-based disks, each entry is 32 bytes long. An entry contains the file name and an index into the file allocation table (Figure 2), which is a group of pointers to the data on the disk device. The directory also contains the date and time the file was created or last modified, the file size, and several attribute flags. These flags show that the file is a system file, read-only, hidden from normal directory searches, and has been modified since the last backup.

Subdirectories are simply special-purpose data files, which can be read and written by both the system and application programs. Many utilities are commercially available for manipulating system directories, performing functions such as alphabetizing entries, sorting entries in date order or file size, or performing other specialized functions.

Like in DOS and OS/2, AIX directories are regular files. However, in AIX, normal programs cannot read or write the directories. A bit in the inode, shown in Figure 7, flags the file as a directory. The directory

entry contains only the file name and its inode number.

Disk Organization

FAT: DOS and OS/2 can handle almost any storage medium and format if a device driver is available. The system needs to know the size of a sector and the number of sectors on the device. The device driver can handle the translations between logical sectors and the physical locations on the device.

In DOS and OS/2, the disk is divided into 512-byte physical sectors, which the operating system logically groups into clusters. Because of a limitation in the addressing scheme used, cluster size is determined by disk size.

For example, on a single-sided floppy disk, the cluster contains only one 512-byte sector. In the PC XT™ hard disk, the cluster contains eight sectors totaling 4,096 bytes. On the PC AT® hard disk, the cluster contains four sectors totaling 2,048 bytes.

The FAT is a structure of pointers to the physical space, or clusters, occupied by a file on the disk. The FAT is located at the front of the disk, just after the boot record.

Pointers have 12-bit values in DOS versions 1 and 2, but were expanded to 16-bit values with DOS version 3 to support the larger disk sizes in the PC AT. This means that there can only be 64 K pointers. With large disk drives, this limitation means that a cluster, which is the unit of space allocation, can be large. Because the number of clusters on a disk depends on the disk size, the FAT size varies with the drive capacity.

File Name	First Cluster	File Size
FILE1.DAT	124	10,000
FILE2.SCR	312	47,021
FILE3.PIC	57	128,745

Figure 2. Simplified Sample DOS-OS/2 Directory Entry

The directory entry for each file has a pointer to the first FAT entry (first cluster) for the file. The first FAT entry points to the next cluster used by the file, and so on, until the last cluster is reached. The system follows the chain of FAT pointers to locate the rest of the data for the file. The FAT is also used to map free space on the disk.

In the simplified directory shown in Figure 2, FILE1.DAT starts in cluster 124 and extends for 10,000 bytes. If the cluster size is 2,048 bytes, this file will use five clusters.

If the disk had many files added and deleted, it is likely that all the clusters for a file will not be contiguous (Figure 3). Assuming that the first four clusters are contiguous, they will occupy clusters 124, 125, 126, and 127. The final cluster is not contiguous because it is located in cluster 131. A sample of the FAT for this portion of the disk is shown in Figure 3.

You can see how the chain of FAT entries for the file FILE1.DAT starts with the entry for cluster 124 and points to each cluster in order. The last cluster, 131, contains a flag hex FFFF, which signifies the end of the chain.

Figure 3 shows the entries for another file, which occupies clusters 128, 129, and 130. There is no way to tell if this file begins at cluster 128, because there could be another FAT entry pointing to that cluster. It is clear, however, that the file ends with cluster 130, because the hex FFFF flag is located in that FAT entry.

Figure 3 also shows how the FAT is used to locate free clusters when the system needs to allocate space to a new file. Clusters 133 and 134 are free as shown by the value of 0 in their FAT entries. Remember that clusters 0 and 1 are always used by the boot record and the FAT itself. The FAT pointer can never have either of these values as part of a valid chain of file allocation pointers.

Because subdirectories are just files, they can also become fragmented. With a cluster size of 2,048 bytes and an entry size of 32 bytes, a cluster can hold only 64 entries. If a subdirectory contains more files, it probably will be scattered across the disk, causing performance degradation if files are frequently used. (Remember that the "dot" and "dot-dot" entries use two of these spaces in the first cluster.)

As shown in Figure 4, each disk has two copies of the FAT. The second copy is for backup if the first one is damaged. The CHKDSK utility uses this copy to correct errors that may occur.

Because the first two clusters are always used by the boot record and FATs, their entries in the FAT have a special meaning. These entries are referred to as "media descriptor bytes," which are used by the system to determine the type of disk being used and the format of the FAT entries.

As previously mentioned, if a cluster's entry in the FAT is not zero, that cluster has been allocated to a file. Free clusters are indicated by having a value of zero in their FAT entry. The boot record, FATs, and root directory are created by the FORMAT command.

FAT Entry											
	124	125	126	127	128	129	130	131	132	133	134
Contents											
	125	126	127	131	129	130	FFFF	FFFF	000	000	000

Figure 3. FAT Entries for FILE1.DAT

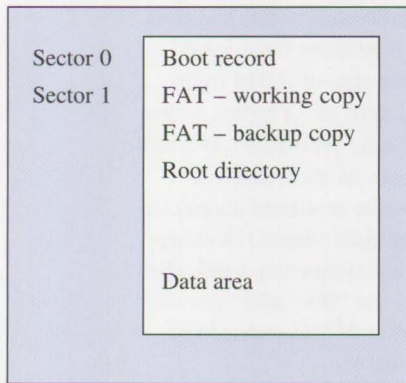


Figure 4. DOS and OS/2 Disk Map (FAT)

Originally, the FAT fit in RAM, giving fast access to the entire disk. With large disk drives, it's not possible to keep the FAT resident, which can cause performance degradation. The system must continually move the head to the front of the drive to find the clusters used when it is reading or writing a fragmented file.

The FAT system was originally designed for single-sided floppy diskettes holding 160 KB of data. While the FAT was once an efficient structure, it is now inadequate for today's high-speed, large-size fixed disks. The new HPFS, which was included as part of OS/2 1.2, is the solution.

HPFS Disk Map: The HPFS disk is divided into areas called bands. A directory band is located near the center of the volume in order to minimize head movement while locating files. There are three fixed control structures in the HPFS: BootBlock, SuperBlock, and SpareBlock.

The BootBlock is larger than the simple boot record in the FAT system. It is larger because the boot program is more complex to handle than the HPFS organization until the full system is loaded.

The SuperBlock contains 32-bit pointers to the major system areas

on the disk including the root directory, a map of bad sectors, a free space map, and other system information. It is modified by system maintenance utilities only.

The SpareBlock contains other system information and is changed as the system runs.

The rest of the disk is divided into 8 MB data areas or bands, each of which has its own free space bit map. In these bit maps, each sector in the band is mapped by one bit, so it is possible to allocate space by sector rather than the larger cluster scheme used in the FAT system. Bit maps for two bands are adjacent, and access is faster because head movement is minimized when files are allocated in adjacent bands.

Data bands repeat as often as necessary to fill the disk. A 32 MB disk, for example, would have four data bands. As shown in Figure 5, it is possible to allocate up to 16 MB of contiguous space to a file organized this way. One of the centrally located data bands stores the directories.

Every file or directory on an HPFS volume is associated with a structure called an *fnode*. The fnode occupies one sector and contains data about the file, including:

- The first 15 characters of the file name
- Extended attributes if they will fit
- A pointer to additional extended attributes
- Pointers to the data
- Access history

For performance reasons, the fnode is stored as close to its file as possible.

The FAT system considers a file to be a collection of clusters on the disk. The HPFS system looks at a file as a collection of runs, or contiguous sectors, of data on the disk. Therefore, instead of mapping each cluster occupied by the file, the fnode contains a pointer to the start of a run and a size or run length field.

The fnode can point to eight runs of data, each of which can be up to 16 MB long. This organization allows small files and contiguous files to be described by a single fnode.

If a file is too large or too fragmented to be described by a single fnode, then the fnode becomes the root of a B+ tree of pointers. In other words, the fnode points to eight other sectors, called *allocation sectors*. These sectors contain the actual pointers to the data, allowing a file to contain many sector runs. Each allocation sector can point to 40 runs.

If this is still insufficient, intermediate levels can be added to the tree, greatly increasing the potential number of runs. Each intermediate level can hold 60 pointers to allocation sectors.

The SuperBlock points to the fnode for the root directory. Like in the FAT system, subdirectories are reached through pointers in their parent directories. Initially, directories are allocated four contiguous sectors. Subdirectories are located in the directory band if there is room. Otherwise, they are located wherever there is free space.

Because HPFS supports long file names, there is no way to calculate how many entries will fit into a directory sector. There is always at least one entry in a directory sector;

if files have short names, there can be many entries. If too many files are to be stored in the 2 KB directory block, a new 2 KB group of sectors is added to the tree structure.

Directories contain a field with the length of the file name, the file name itself, date and time stamps, and other control information. This information may be the directory entry length or a pointer into the B+ tree structure.

File names are stored in the directory in alphabetical order. Therefore, directory searches are generally faster than in the FAT system because it is not necessary to search the entire directory to locate a file. When the first file later in the alphabetical sequence is reached, the system knows that the file does not exist and stops looking for it.

If you convert to HPFS and do not change the file name size, about 65,000 files can be listed in a three-level directory tree. This means that in three disk accesses, you can locate a file or prove that it does not exist. The same directory size under the FAT system would take more than 4,000 disk accesses to prove that the file did not exist.

However, there is a downside. While locating a file is quite efficient, it can take a long time to add a new file to the directory or to rename an existing file. A new name must be added at the proper location in the B+ tree structure. The entire structure may be reorganized if the directory block is full and a new level needs to be created to make space for the file.

Partitions: There is a more fundamental difference between the DOS and OS/2 file systems and the AIX file system. In DOS and OS/2, each

Sectors 0 - 15	BootBlock – boot code
1 sector	SuperBlock
1 sector	SpareBlock
Data band 1	
	8 MB data area
	Free space bit map for band 1
	Free space bit map for band 2
Data band 2	
	8 MB data area
Data band 3	
	8 MB data area
	Free space bit map for band 3

Figure 5. HPFS Drive Layout

disk device has a unique logical identifier assigned to it. For example, the first floppy disk drive is always known as drive A:, the first fixed-disk as C:, and each drive is assigned the next available letter as its ID as it is located by the system during the boot process.

As described earlier, each logical drive has a self-contained directory structure. It is independent of all other drives in the system. Valid disk names, or identifiers, are the letters A through Z, limiting DOS and OS/2 to a maximum of 26 disk devices. New drives cannot be added without rebooting the system.

With DOS and OS/2, fixed disks can be partitioned into multiple smaller logical drives. Each of these smaller partitions receives its own unique drive identifier. For example, if an accident, like the corruption of the FAT or directory, destroys the structure of a logical drive, it will not be necessary to rebuild and reload the

entire fixed disk. Rather, only the affected partition would need to be rebuilt.

Under OS/2, it is possible to intermix both FAT and HPFS logical drives in the various partitions on the fixed disk. If compatibility with DOS is desired, the HPFS partitions should be the last partitions on the disk. These partitions will not be recognized by DOS, and the higher drives' ID letter will be different when OS/2 is running. This process could be very confusing.

AIX Disk Structure

In AIX, the disk is organized differently than the FAT-based organization of DOS and OS/2. Directories contain only a file's name and a pointer to a structure called an *inode*. All of the information about the file, including allocated space, date and time stamps, and so forth, is contained in the inode. The inode is similar in concept to the HPFS *fnode*.

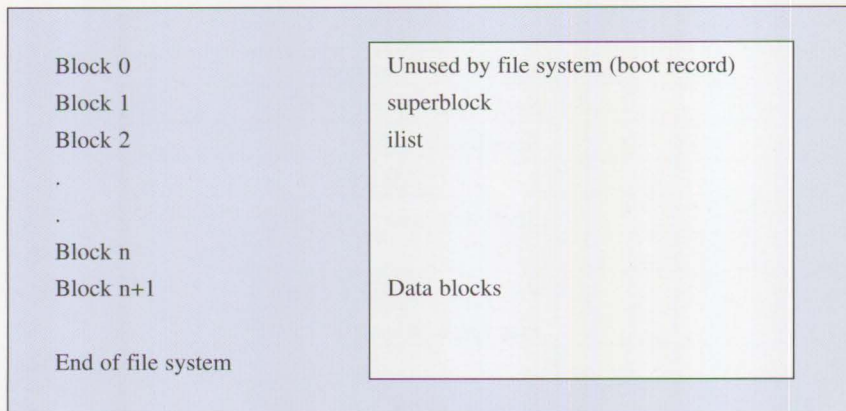


Figure 6. AIX File System Map

Minidisks: In AIX, all disks form a single logical structure, which is divided into minidisks. The AIX system can support up to 128 minidisks, with up to 32 minidisks on any physical disk drive.

The AIX file system is called a mountable file system. A mount point is an empty directory in the file system, which means that a new disk may be added to the logical file structure at any time. This is significantly different than DOS and OS/2.

Each time the AIX PS/2 system is started, it checks to ensure that the file system structure is intact. If problems are found, the system tries to repair the damage, although it is not always successful. This is why the shutdown command must be executed before the system is turned off.

It is sometimes necessary to reformat the fixed disk when installing a new version of the system. Reformatting would take advantage of a new feature or just ensure that the disk has no remaining files from the previous system. With DOS and OS/2, only the C: partition needs to be affected by a reformatting, leaving the user data on the other logical drives unaffected. With AIX PS/2, it may be necessary to unload the en-

tire file system if the previous minidisk allocations did not leave sufficient room for the new system.

While several minidisks are created when the AIX system is installed, it is often necessary to create additional minidisks. Space can be reserved for this purpose during installation, or an additional fixed disk may be added to the system. The minidisk commands can be used to create one or more additional minidisks from this free space.

The AIX disk is allocated by 4,096-byte logical blocks, each made up of eight, 512-byte physical blocks. A map of the file system resembles the diagram in Figure 6.

AIX Disk Organization: Block 0 contains the boot record, but it is not used by AIX PS/2. A separate bootable partition, which actually boots the system, is created on the hard disk.

The superblock tracks the file system size and name, the number of blocks reserved for inodes, and the free block list. While the system is running, the superblock is kept in memory. Before the system can be turned off, this data must be written to the disk. AIX systems require a

special shutdown command to be issued before the system is turned off. (The OS/2 system has the same requirement if the HPFS file system is used.)

The *ilist* contains structures mapping a file to the physical blocks on the disk, much like the FAT does in DOS and OS/2 systems. Like the FAT, the size of this area depends on the file system size. Each structure, which is 64 bytes long, is called an *inode*. There is an inode for each file. Inodes are numbered sequentially from 0 to the maximum that the file system can hold and are indexed by this number. Index 2 generally is the root directory.

Inodes contain information about the file they refer to, such as:

- File mode
- File type
- File length
- Owner and group ID numbers
- Date and time stamp
- Number of links
- Data blocks physical location

Each inode contains 13, four-byte addresses, as shown in Figure 7. The first 10 addresses point to data blocks in the file.

If the file is larger than can fit in these 10 blocks, indirect block pointers are used. These are blocks pointed to by the inode that contain pointers to the file's data blocks. The number of data blocks that can be pointed to by the indirect block is dependent on the disk block size, which ranges from 128 bytes on diskettes to 512 bytes on fixed disks.

Even larger files may need to use block 12 that points to a second level of indirect pointers. Block 13

points to a third level of indirect pointers for very large files.

The system keeps track of unused inodes created when a file is deleted, so they can be reused. The superblock contains an array of free inodes, a count of free inodes, and a count of the total number of inodes in the system.

The last part of the file system is the data area. It contains the data files, the indirect index blocks for large files, and the free blocks.

The file system also maintains a list of free data area blocks in a free block list, which is a linked list of

pointer blocks. These blocks are used for data files, indirect index pointers, and so forth, as needed. The superblock contains pointers and counters to maintain this list.

Each free block pointer contains pointers to up to 50 free blocks, as well as the link pointers to the next free block pointer in the chain. Blocks are added or removed from this chain as needed to track the free blocks in the file system.

AIX File Types: There are four file types in the AIX system: directory, ordinary, special (character, block, FIFO, and sockets), and symbolic links (which contain pointers to other files).

Directory files are, as the name implies, subdirectories containing lists of filenames and associated inodes. Ordinary files are regular data and program files. Special files are used to run the system. These may not exist in the file system, because they are really pointers to specific device drivers. Finally, symbolic links are files containing pointers to other files. These pointers enable a user to give a short, meaningful nickname to a file.

Special Directories: The AIX system uses certain directories for special functions. The /bin directory stores many user commands. All user passwords are kept in the /etc/passwd file. The /usr directory

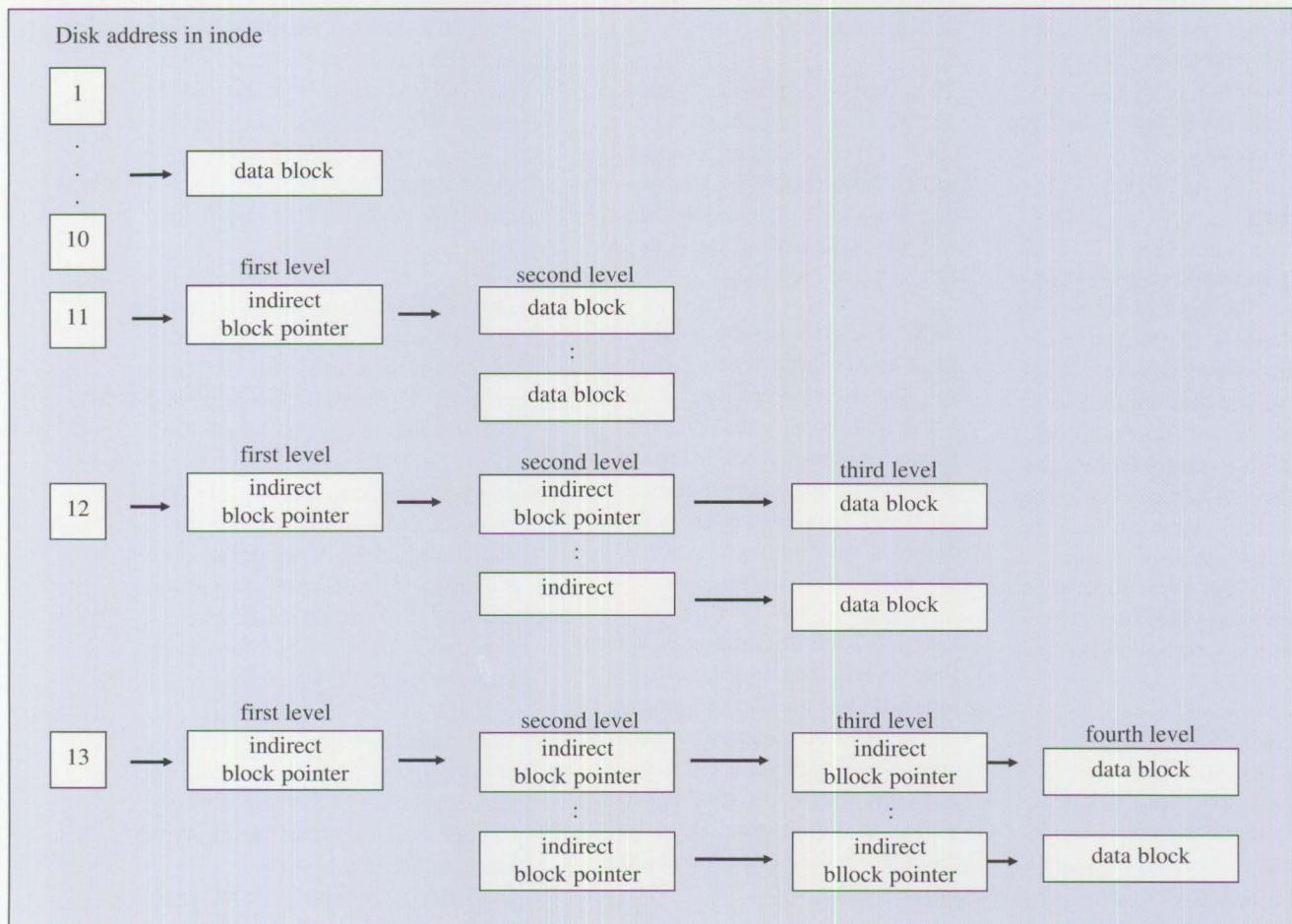


Figure 7. AIX Inode Data Structure

contains subdirectories associated with print spooling, mail, utility commands, and others.

Security has been mentioned briefly. The `/usr/adm/sulog` file logs all user attempts to issue the `su` or switch user (superuser) command. Because this command gives users system administrator capabilities, all attempted uses of this command are logged, including the name of the user, date and time, terminal used, and so forth.

Of course, there are many more special directories in an AIX system. This topic was only mentioned to introduce the concept of special reserved directories. DOS has no reserved directories, and OS/2 has just a few. For example, OS/2 Standard Edition has just one, `\OS2` and its related subdirectories. These directories contain all of its code except the loader and kernel, which are in the root directory.

Installation

DOS: As you would expect, DOS is the easiest of the three systems to install. DOS 4.00 is shipped on two diskettes. Experienced users can install DOS by copying the files to a directory on the hard disk and then running the `SYS` command, which copies the hidden kernel files to the target disk's root directory. Later releases include the `SELECT` command that displays a few screens of questions, allowing new users to specify how the system will be configured.

OS/2: This operating system is more difficult to install because it is substantially larger than DOS. It is packed as compressed files on eight high-capacity diskettes for Standard Edition version 1.2. It is necessary to use the installation diskette because the files are compressed and

there are so many files whose functions are undocumented.

The installation diskette is booted and loads a restricted version of the OS/2 system. Several screens are displayed, allowing users to tailor the system to their requirements. The installation program then creates several subdirectories into which the uncompressed files from the remaining seven diskettes are copied.

Version 1.3 adds additional flexibility to the installation process. There is an additional menu that lets users select the components to be installed. This allows users with limited free disk space to eliminate system functions not needed, such as the tutorial and online documentation.

The installation process is not too complicated for Standard Edition, but it is more difficult for OS/2 Extended Edition. OS/2 Extended Edition has a Communications Manager and a Database Manager with separate installation menus.

AIX: The AIX system is also quite large and is shipped on 19 high-density diskettes for version 1.2. Several blank high-density diskettes are needed, because at least three of the diskettes must be copied during installation. More diskettes may be required if system maintenance is installed.

Like OS/2, an installation diskette is first booted. Three of the distribution diskettes are used for output, and need to be copied to other diskettes. From a menu, users can format and copy the distribution diskettes written to during installation, thus eliminating the need for changing original diskettes.

The system is then rebooted, and the installation process begins. AIX requires considerably more information for installation than either DOS or OS/2.

Some installation questions will surprise former DOS users. For example, with AIX there is a requirement to enter your time zone, and whether or not you are on daylight savings time. Even your machine needs to be named. (The name is used if your system is on a network. Time zone information converts time stamps from other systems to your local time.) These questions are not difficult to answer.

AIX also has special machine requirements. If you are using a PS/2 with an enhanced small device interface (ESDI) drive, you may get a message indicating that the disk controller ROM needs to be upgraded. If one appears, there is a no-cost engineering change (EC) that your customer engineer will install for you.

During installation, the fixed disk is divided into several minidisks, which are then formatted into file systems. The installation process creates the root, user, temporary, and local file systems, a virtual memory paging space, and a dump file. The sizes for these minidisks may be specified instead of using the system defaults. You can also reserve some disk space to use later for new minidisks. The distribution files are then copied to the hard disk.

With both DOS and OS/2, there is nothing in the installation process that cannot later be changed easily. Changes can be made by simply editing the configuration file, `CONFIG.SYS`, which is located in the root directory. With AIX, however, the file system space is allo-

cated to various system functions and minidisks. If this is not done correctly, it may be necessary to reinstall the entire system.

Programming Considerations

The Intel 808X family of microcomputers is interrupt-driven. In other words, when one of a list of events occurs, the processor interrupts what it is doing and jumps to a new section of code. The various events that can cause interrupts are assigned a unique number. The address of the code handling that function is stored at the beginning of system memory in the interrupt vector table. The code address for each interrupt is the interrupt number times four, which is the size of an address pointer in the table.

For example, when a key is pressed, the CPU is interrupted by the keyboard interface hardware. The CPU must save its status at the time of the interrupt and branch to the instructions that handle the data being sent by the keyboard. The CPU must then reload its saved status and return to the instructions it was processing before the key was pressed. The CPU can be interrupted by either hardware or software through with a special interrupt (INT) instruction.

Because DOS is a single-tasking system, programs assume they own the entire computer. DOS was designed around the system interrupt capability. When a program wants DOS to perform a function, it does not call DOS directly. Rather, the program executes a software interrupt instruction. DOS uses interrupts 20H through 26H.

Programs often change system interrupt vectors to point to their own code. As a result of this change, pro-

Handle or Descriptor	DOS	OS/2	AIX	Device
0	stdin	stdin	stdin	keyboard
1	stdout	stdout	stdout	screen
2	stderr	stderr	stderr	error
3	stdaux			auxiliary
4	stdprn			printer

Figure 8. DOS, OS/2, and AIX Handles

grams can perform simple multitasking, extend the functions provided by DOS, and so forth. For example, the typical pop-up program changes the keyboard interrupt vector to point to itself. The program activates itself upon detecting its hot-key combination. Other keys are passed to the original DOS keyboard code.

Because OS/2 and AIX PS/2 are multitasking systems, they cannot operate in this manner. If one program was allowed to take over a hardware interrupt in a multitasking system, the other programs or the system itself would probably fail, thereby risking data loss or corruption. For this reason, neither OS/2 or AIX PS/2 allow programs direct access to the interrupt vector table.

Regular programs are not allowed to execute software interrupts or to change shared memory without system control. A call-based interface is used instead of the software interrupt interface used by DOS. The routine that is called can check the validity of the request before passing it to the operating system for processing.

In OS/2, programs can intercept data from the keyboard or other devices under system control by using a special program called a "monitor." The system gets the data from the device and passes it to the monitor for processing. Then it passes the changed data through the normal system

code, eliminating the need for programs to intercept the hardware interrupts.

When porting applications to either OS/2 or AIX PS/2 from DOS, all system calls must be recoded. The interrupt instructions must be changed to calls. If a high-level language is used, the compiler will make these changes. If Assembler or low-level C calls are used, the program changes could be extensive.

In all three systems – DOS, OS/2, and AIX – several handles are assigned when a program is started (Figure 8).

It is common practice in AIX programs to close handles 3 through 20 during program initialization. This should not be done in DOS and is not recommended in OS/2 as some of these handles may have been opened by dynamic link routines.

AIX has one major advantage for programmers – a flat address space. The Intel microprocessors used in DOS all have "segmented" memory. This means the largest contiguous amount of memory that can be addressed is 64 KB. To maintain compatibility with DOS and the 80286 microprocessor, OS/2 continues this segmented approach to memory management.

Beginning with version 2, OS/2 adopts the flat memory addressing



scheme used by AIX. This is a tremendous advantage for programmers who are working either with large programs or large amounts of data. With this addressing scheme, programmers no longer have to keep track of which 64 KB segment contains the code or data being used. The flat address space eliminates the need for the large number of memory models used by languages like C. All programmers benefit from this change.

System Calls: To a programmer, a system's power is proportional to the number of function calls it provides. IBM technical reference manuals list 99 system calls for DOS, 265 for the OS/2 kernel, and 550 for the Presentation Manager, and 66 for AIX PS/2. (Note: The AIX number

looks low, but the AIX system calls look like C verbs and are difficult to count.)

Multitasking

DOS: There is no documented multitasking support in DOS. However, it is possible for a program to terminate without releasing its memory. A DOS program can give the appearance of multitasking by taking over system interrupts and terminating while keeping its memory.

For example, if a DOS program takes over the keyboard interrupt, it can activate itself, or pop up, when a user presses a specific hot key. While this is really not multitasking, it does give that appearance to the user.

DOS programs also can take over the timer interrupt, which occurs many times a second. This gives the program the opportunity to execute a small amount of code before passing control to the original interrupt handler. This is a dangerous type of multitasking. There is no system control over how the CPU time is allocated or over the number of programs that take over a particular interrupt.

As discussed previously, DOS real mode programs assume they are the only program running in the system. Therefore, they are free to use any memory or hardware in the system. If other programs are running, there could be total havoc, as each program overlays the memory of another, writes to the same file, or sends data to the printer. If one of these programs should crash, it will probably bring down the entire system, causing loss of data when other programs are stopped.

To the standard definition of multitasking, which is the ability to run multiple programs simultaneously, let's add the phrase "under system control." The new definition for a multitasking system is "a system that allows multiple programs to execute simultaneously under system control."

Several programs add multitasking capability to DOS. These include Microsoft Windows Version 3 and Desqview. On the 80386 and later processors, these programs take advantage of the virtual 8086 mode to isolate and protect each DOS program.

OS/2: A time-sliced, priority-based, pre-emptive scheduler is used in OS/2. In fact, OS/2 has many multitasking capabilities that are found on large mainframe systems. This

means that an OS/2 system can run more than one program at a time, and the system controls these programs. There is no chance that one of these programs will take over the entire system. As will be covered later, OS/2 has greater flexibility in multitasking than the AIX PS/2 system.

The unit of resource allocation or ownership in both OS/2 and AIX is the "process." Program, task, and process are often used interchangeably.

By default, each process (or task) has the keyboard, display screen, and some memory allocated to it. It is possible to start a background process that does not always have access to the screen and keyboard. Each process usually has other resources allocated such as data files. A process should be thought of as a copy of a program loaded into system memory plus the resources used by that program.

Resources are allocated to a process. The CPU is allocated to a thread. If a process is thought of as a complete program, then a thread is equivalent to a function within that program. Each process has at least one thread. The thread is really just a stack area, an entry in a system table containing the thread's ID and status, and a pointer to the next instruction to be executed by the CPU for that thread. A thread can concurrently execute with other threads within the same program or process.

If a process needs to do more than one activity at a time, it can create another independent process to perform the second activity. For example, if a system has two communication ports, it may be desirable to start a second process to handle the second port. In this case,

these would be completely independent tasks in the system. Each process would own one port, and the code executed by each of these processes would be an independently scheduled thread.

The new process is created with the `DosExecPgm` system call. The system loads and executes the specified program as either a child process or as a completely independent process.

In fact, OS/2 has many multitasking capabilities that are found on large mainframe systems.

There are times, however, when a program needs to do more than one activity at a time, but does not want to create an entirely independent process. For example, a person is using a word processor. The program is loading a large document file for updating by the user. The program may start reading the document. It may then stop reading long enough to show the first screen full of data before continuing to read the rest of the file into its buffers.

This sequence could be done in DOS by counting the lines of text in the document as they are being read. When a full screen's worth of lines have been read, that screen of text is displayed before the rest of the file is read.

In OS/2, there is a better way to do this. As previously stated, programs are divided into threads of execution. The typical program has just

one thread. In this sequence, two threads are needed: one to read the document file and another to display the text.

The thread reading the file sets a flag when enough data has been read to fill the screen. The second thread will then begin to display the document while the first thread continues reading the remaining data. OS/2 provides a special flag, called a semaphore, just for this purpose. Semaphores, which are a form of interprocess communication or synchronization, are discussed in the section titled "Semaphores."

If the user scrolls ahead in the file, the display thread can check the buffer. If enough data has been read, the thread can display the next screen of text even though more data remains to be read. The result is to simultaneously read and display data, which makes the system seem more responsive to the user. If the user then decided to check the spelling of a document, the program could use an additional thread to perform the spell check while the user is editing the document.

The system can create a new thread faster than it can create a new process. This is because the system doesn't have to perform any resource allocation. The thread has less status information associated with it; therefore, the system can switch between threads faster than it can switch between processes. Each thread shares memory with all other threads in the same process, which makes it easy for them to communicate with each other. This makes threads a high-performance multitasking mechanism. Generally speaking, processes are the way the CPU is shared between applications, and threads are the way the CPU is

shared within a single application program.

AIX: OS/2's multitasking capabilities were described as being similar to a mainframe system. AIX PS/2, being based on UNIX and very similar to the System/370 version, is very close to a mainframe system with all the capabilities that implies.

In AIX, multitasking is limited to running multiple processes. AIX does not have the concept of threads. While this allows for multiple programs to execute concurrently, it reduces the ease of achieving multitasking within a single program, thereby complicating the task facing application programmers.

In AIX PS/2, the fork and exec functions are used to start a new process. Unlike in OS/2, a new program file is not loaded into memory. Instead, a second copy of the parent program is made in memory. Both the child and parent process begin executing with the instruction following the fork call. It is up to the two processes to test their process IDs, decide whether they are parent or child, and branch accordingly. The child will start its new process with the exec function call.

AIX has one handy function that is missing from OS/2. The "kill" command immediately terminates the specified process.

In the last OS/2 example, there was a single process with two threads reading and displaying data simultaneously. To do this without threads would be more difficult. In both OS/2 and AIX, all resources are allocated to a process, including system memory. Using one process to read data into a memory buffer, and a second process to display the data from

that buffer, forces the programmer to find a way to share that part of memory between the two processes. Shared memory would be used to do this. This is discussed next as one means of interprocess communication.

Interprocess Communication – DOS and OS/2

DOS, like OS/2 and AIX PS/2, can "pipe" or pass output from one program to another as its input as they execute sequentially. This section is about how processes can communicate as they execute simultaneously.

Because OS/2 is a multitasking system, there are several ways for its processes to communicate. These range from simple to complex. The following discussion looks at these methods in order of increasing complexity.

Pipes: Used on the command line, pipes are the most common way to pass data between different programs. Pipes tell the system to take the output from one program and redirect, or pipe it to another program as that program's input as is done by DOS. With pipes, a FIFO stream of data is sent in one direction between any processes in the system. Pipes are slower than shared memory because the data is first passed to the system by the sending program, and then passed from the system to the receiving program. They are also limited to a 64 KB data segment size. If two-way communication is needed, then two pipes must be used; one for reading and one for writing. Piped data is kept in memory, it is not written to disk.

OS/2 added support for named pipes in version 1.1. Named pipes allows a pipe to send data between unre-

lated processes, including those on different OS/2 systems if connected on a network.

The pipe looks like a regular file, but with special naming conventions. The system takes care of moving the data to the receiving process on the receiving system. OS/2 Extended Edition must be used if a named pipe is used between different systems.

Named pipes can be used to send and receive either bytes of data or messages. Therefore, with a single call, a process can receive a complete message from another process.

Shared Memory: Using shared memory is the fastest way to exchange data between processes. It is necessary for each process to issue the proper system commands to have access to the shared memory area. Shared memory also has the greatest flexibility for the applications programmer, because the data in the shared memory can be arranged any way the programmer desires. Also, with shared memory, data can be randomly accessed.

Queues: These are the final form of interprocess communication. This is the most versatile form, because data can be any length and can be read in both a FIFO or LIFO (last in, first out) order, or by message priority. Queues are useful when one process collects data from several other processes; for example, a data collection program.

Data Files: These can also be used as interprocess communication. While slower than the other methods, files provide a recovery capability in case the system should crash. They also allow for large quantities of data to be shared.

Semaphores: These were briefly mentioned earlier. Semaphores synchronize different threads or processes. One thread waits to perform some function until it is notified by a semaphore that there is work to be done or that a needed resource is available. While waiting for the semaphore, the system puts the thread to sleep. OS/2 semaphores are binary, which means they are either set or clear. This is clearly a form of interprocess communication. Because only a ready or not-ready signal is being communicated, it's not a method of sharing data.

OS/2 has two types of semaphores – system and RAM. System semaphores are maintained by the system and can be used by any process in it. They follow a naming convention similar to regular files. A RAM semaphore is not maintained by the system. It is simply a global variable within a program. Therefore, it can only be used by threads within a single program.

Interprocess Communication – AIX

In AIX, there is the same support for pipes as in OS/2 plus some additional capabilities. Data can be redirected on the command line using similar command syntax, but with the tee command, data can be redirected to two places simultaneously. In other words, AIX PS/2 not only can pass results of one program to another, but it can also display or print the same information or route it to a third program.

AIX pipes can only be used between related processes; that is, between parent and child processes or between child processes with the same parent process. As in OS/2, pipes are FIFO data streams.

While AIX PS/2 does not have a named-pipe function, it does have a special file type called FIFO. Its functions are similar to the named pipe in OS/2. Because it is a file, it is stored on disk. FIFOs do not require that processes using them are related; in fact, the print spooler is implemented this way.

Both message queues and semaphores are similar to their equivalent OS/2 functions. Message queues are used to share data, and semaphores are used to synchronize processes. In AIX, the entire process is put to sleep if the semaphore it is waiting for is unavailable. This differs from OS/2 where the CPU is dispatched at the thread level. In OS/2, semaphores are referenced by name. In AIX PS/2, they are referenced by an ID number.

In OS/2, a semaphore is either set or clear. In AIX PS/2, a semaphore is really a counter and can have many values. For example, if a resource can be used by several processes at one time, the semaphore could be a count of the number of active users. Instead of testing to see if the semaphore is just set or clear as is done in OS/2, the AIX program tests the semaphore for the actual count of users of the resource and branches based on the result.

A group of semaphores that are created or manipulated with a single cell are referred to as semaphore sets. These sets are provided by AIX PS/2.

AIX PS/2 has a shared memory function similar to the one in OS/2. As with OS/2, this function provides the fastest way to share data, because it is not moved among the different processes using it.

Signals: Another form of inter-process communication is the signal. Usually used to communicate the occurrence of an external event, an example of a signal is a user pressing a key to abort the program (Ctrl-Break in DOS and OS/2, Ctrl-C in AIX). One process also can send a signal to another process. If a process wants to handle a specific signal, it must establish a routine to do so. Otherwise, the system will take default action for the signal, which is usually to terminate the program.

Device Drivers

A device driver is a program that interfaces between operating system calls and a particular hardware device. Device drivers are designed to shield the system kernel from handling device operation details.

For example, because DOS supports floppy disk drives, fixed disk drives, the screen, keyboard, and a printer, it includes specific routines in the kernel that provide the programming interface to these devices. However, if a programmer knows the details of the device function calls, a cleverly written program could directly address a specific device, thereby bypassing the built-in device driver.

If a new type of hardware device is added to the system that is not supported by the built-in device drivers, the appropriate device driver must be added to the system. This device driver is usually supplied by the hardware vendor.

OS/2 includes the same device drivers found in DOS as part of the kernel. Because the kernel contains drivers for most standard hardware, on DOS and OS/2 systems the VDISK.SYS driver is the most commonly added device driver. VDISK.SYS allocates and organizes

a portion of system memory as a logical disk drive, giving programs fast access to small amount of often-used data.

In both DOS and OS/2 systems, device drivers can be located anywhere in the file system. They can be included in the system by adding a DEVICE=driver file name line to the CONFIG.SYS file. This file is read during system initialization and identifies device drivers to be loaded. If a driver specified in the CONFIG.SYS file works for one of the default system devices, the standard driver in the kernel is not used.

OS/2 device drivers are considerably more complicated than their DOS counterparts. The complication results from OS/2's multitasking support. This support often results in more than one outstanding request. Because OS/2 operates in both real mode and protected mode, device drivers must be bimodal. In other words, they handle both physical addresses for buffers when operating in real mode and virtual addresses when in protected mode.

OS/2 drivers also must be able to remove themselves on demand, which is done to conserve memory if multiple drivers for the same device are loaded. The previous driver can release its memory because its functions are handled by the second driver.

With these special OS/2 considerations, OS/2 device drivers must be re-entrant. This is a more difficult way to write a program. OS/2 has special functions used by device drivers called Device Driver Helpers, or DevHlps, which make this an easier process. Beginning with OS/2 Version 1.1, special interdriver communication (IDC) entry points were also established.

The AIX system is considerably different because programs never directly address a device. The AIX device driver is the only code that talks directly to a device. There is a device driver for every device in the system. Device drivers must be linked into the system using the **ld** command. To add another device driver, the system must be generated (linked) and then restarted.

Device Monitors

Some DOS applications permanently load into memory and then activate themselves when an operator enters a certain "hot key" sequence. Referred to as a "pop-up" or terminate and stay resident (TSR), this type of program and has become quite popular. Borland's SideKick is an example of this type program. Unfortunately, this type program takes over a key system resource – the keyboard (and perhaps the timer) – without the system having any control over it.

As these programs proliferated, their combined memory requirements became significant. There was often a conflict between their hot keys, timer interrupt usage, and so forth. Standards have been proposed to control how these programs operate, but they are not universally followed. The optimum solution is OS/2's device monitor capability. A device monitor is a type of program inserted into the layered system structure at the device-driver level.

Device drivers are difficult to write because they work closely with the hardware, yet device monitors are easy to write. The device driver continues to control the actual hardware device. A monitor signals that it wants to look at incoming data from a specific device. The device driver passes each data item to the monitor after it has been read from the device, but before it is passed onto the system. Monitors can look at output data, too.

Command	Function
CALL	Calls another batch file from within a batch file
ECHO	Displays or suppresses echo of commands as batch file executes; or echoes this command
FOR	Repetitively executes a command
GOTO	Branches to a new location in the batch file
IF	Tests a condition and branches if true
PAUSE	Stops execution of batch file, displays user prompt, and waits for a keystroke
REM	Defines a comment line

Figure 9. DOS and OS/2 Batch File Commands

Command	Function
ENDLOCAL	Restores the drive, directory, and environment variables in effect when the previous SETLOCAL command was issued
EXTPROC	Defines an external batch file processor
SETLOCAL	Saves the drive, directory, and environment variables currently in effect to restore later with an ENDLOCAL command

Figure 10. OS/2 Batch File Commands

This device driver function isolates the monitor from the hardware while including it in the data path. Multiple monitors can work with data from the same device if chained together by the device driver. Data is then passed to each of the monitors in turn. A single monitor can register with several device drivers and act as a path between them. These programs could be print spoolers, keyboard enhancers, or other utilities that filter data independently of other programs.

OS/2 includes a special monitor dispatcher to control device monitor execution. Because the monitors operate at such a low level in the system architecture, they must be designed carefully not to adversely affect system performance.

Because the monitor is controlled by the device driver, the device driver must include the necessary support code for this function. The standard OS/2 device drivers for the keyboard, mouse, and printer have the necessary support code. However, the COMx and SCREEN\$ drivers do not have this support for the screen and serial devices.

Shell Programs, Batch Files, and REXX

Figure 9 lists the special batch file commands available with DOS or in the DOS compatibility session of OS/2.

When not using REXX while in protected mode, OS/2 adds three additional commands to those available under DOS. Figure 10 lists these commands.

The only special variables available to DOS and OS/2 batch files are %0 through %9. These variables were

Name	Purpose
BEEP	Sounds the system speaker
CALL	Allows internal routines (labels), built-in functions, and external functions to be called as subroutines
CHARIN	Reads characters in from an input stream
CHAROUT	Writes characters to an output stream
CHARS	Returns the number of characters remaining in an input stream
COMPARE	Returns the result of the comparison of two strings
COPIES	Returns concatenated copies of a string
C2D	Converts a character to decimal
C2X	Converts a character to hexadecimal
DATATYPE	Returns the result of checking the data type
DATE	Returns the date set on your PC
DELSTR	Deletes a substring as a character unit
DELWORD	Deletes a substring as a word unit
DIRECTORY	Returns the name of the current directory
DO	Groups instructions and optionally runs them repetitively
D2C	Converts decimal to character
D2X	Converts decimal to hexadecimal
EXIT	Leaves or ends a program unconditionally
FILESPEC	Returns a selected file specification of either drive path or name
FORMAT	Rounds and formats a number
IF	Allows conditional processing of functions, instructions, and OS/2 commands
LASTPOS	Returns the position of the last occurrence of a string
LENGTH	Returns the length of a string
LINEIN	Reads a line from an input stream.
LINEOUT	Writes a line to an output stream
PARSE	Assigns data from various sources to one or more variables
PULL	Reads a string from the head of the external REXX data queue
RETURN	Returns control, and possibly a result from a REXX program or internal routine, to its starting point
SAY	Writes to the default output stream
TIME	Returns the local time set on the system in the format hh:mm:ss or in the format specified in the option
TRACE	Selects or sets the tracing of system events

Figure 11. Selected REXX Commands

Command	Function
for	Executes a command repetitively
if	Tests a condition and takes action as appropriate; action can be complex group of commands; if the tests are false, and an else branch is provided, the else commands are executed
while	Executes group of commands while tested condition is true
until	Executes group of commands until condition is true
(list)	Executes command in the list in a subshell
{ list; }	Executes command in the list in the current shell
:	Returns a 0 exit value; null
.file	Reads and executes the commands in file in the current shell
break	Exits from a for, while, or until loop
continue	Resumes execution of a loop
echo	Displays command line
eval	Reads arguments as input to the shell and execute the resulting command
exec	Executes specified command in place of this shell in this process
exit	Leaves shell and sets a return code if specified
export	Marks specified names of environment parameters for passing to subsequently executed shells
hash	Remembers location in the search path for each specified name
newgrp	Changes primary group identification
pwd	Displays current directory
read	Reads a line from standard input and assigns each word to the specified variables
readonly	Makes specified variables readonly so they cannot be reset
return	Exits a function and passes a return code if specified
set	Sets flags controlling script execution
setxvers	Sets experimental version prefix to the specified string
shift	Moves command line left logically by decrementing the index of each parameter
test	Evaluates conditional expression
times	Displays accumulated user and system times for processes run from shell
trap	Runs specified command when system sends specified signal to shell
type	Indicates how shell would interpret specified command name
ulimit	Sets or queries size limits
umask	Sets user file creation mask to specified value
unset	Removes corresponding variable from the environment for each name specified
wait	Waits for specified child process to end

Figure 12. Shell Script Commands

\$\$	Process ID
\$#	Number of parameters on command line
\$*	Text string with all command line parameters
\$?	Last return code from executing program
\$!	Process ID of last process started in background
\$-	String with execution flags currently set in shell
\$1 . . . \$9	Parameters passed on the command line

Figure 13. Variables Passed into Shell Scripts

passed to the batch file on the command line. If more than nine were passed, the SHIFT command must be used to address the additional variables. Environment variables can also be specified as replaceable parameters.

REXX has several other major capabilities. A program can use it to execute a REXX program or to receive and execute REXX commands. Programs written in other languages can be invoked as REXX functions, and programs can read and write REXX variables. In other words, REXX can control multiple programs running in OS/2 and can pass data and commands between them by controlling the OS/2 system. Figure 11 lists some REXX commands.

The AIX PS/2 system has similar power to REXX. Its command lists are called shell scripts. Like REXX, shell scripts can get input from the user, which can then be used to control the script program's execution path. Shell scripts are so powerful that it is not uncommon for applications to be prototyped as shell scripts before the actual code is written. There is even a commercial bookkeeping package available that is written as a shell script.

File names are not used to tell the system a file can be executed. After creating the shell script, the chmod

command must be used to mark the script executable before it can be run.

AIX PS/2 has many shell script commands. Each shell has a different set of commands and syntax. The Bourne shell is the default for AIX PS/2. The commands for the Bourne shell are listed in Figure 12.

AIX PS/2 makes extensive use of the environment to support shell scripts. Figure 13 contains a list of the standard environment variables passed by the Bourne shell to shell script programs.

Summary

Part 2 has presented a more technical description of the DOS, OS/2, and AIX PS/2 operating systems to help you understand how these systems are designed. The programming capabilities and installation procedures have also been discussed.

We can conclude that DOS is the operating system of choice for users with little technical knowledge and with limited programming requirements. Microsoft Windows Version 3 gives the appearance of the Presentation Manager with a limited form of multitasking, which gives you time to wait for OS/2 Version 2.

It is relatively easy to migrate to OS/2 from DOS. Programs do not

have to be converted to OS/2 functions immediately, because the DOS compatibility session allows most existing DOS programs to continue being used. Microsoft Windows can still be run in the DOS compatibility session, which will make migration to Presentation Manager very easy. OS/2 2.0 will run Windows applications directly, thereby preserving your software investment.

As the most powerful and complex of the three systems, AIX PS/2 is, in my opinion, the ideal system to use in a network. It is designed for multiple users who share the same computer or who need compatibility or connectivity with UNIX systems. AIX has more functions at the command interface than either DOS or OS/2. If you have a requirement for multiple users to connect to the same system unit, then AIX PS/2 is the perfect system.

ABOUT THE AUTHOR

W. Craig Chambers is a senior market support representative, supporting RISC System/6000 and AIX in the Engineering/Scientific National Support Center. Previous assignments include lead systems engineer at several large MVS/JES2/JES3 accounts and technical support for OS/2 SE and for communications products, including the 3270 system, 3270-PC, the Workstation Program, and Enhanced Connectivity Facilities. He has been a presenter for IBM's Technical Coordinator Program television broadcasts and has published several IBM technical bulletins. Craig received B.S.M.E. and M.S.M.E. degrees from Purdue University.

Using Dual Displays with OS/2

Larry Pollis
IBM Corporation
Dallas, Texas

Traditional support for dual displays cannot be achieved on PS/2 systems. This article shows what can be done under OS/2 and explains the differences in behavior of dual displays on PS/2 systems and its predecessors.

With the release of OS/2 Version 1.2, support for dual displays has been provided by the operating system on the PS/2 line of computers. This support has been implemented within relatively narrow constraints, which are carefully defined in this article to avoid any misunderstandings.

Prior to PS/2 systems, dual display support could be set up by installing a monochrome display adapter and a color display adapter in an IBM PC, XT, or AT system (or compatible). Because of how PCs address video RAM, dual display support can only be obtained with one adapter of each type. The color adapter could be CGA, EGA, VGA, or the Professional Graphics Adapter.

Applications had to be written with dual display capability in mind to utilize both displays. Three applications that include this support are Lotus 1-2-3™, AutoCAD®, and CodeView™. Both Lotus 1-2-3 and AutoCAD used one screen for the text-based information and the other for displaying graphics (a chart with Lotus 1-2-3 and a drawing in AutoCAD). CodeView used one screen to display the program source code, breakpoints, variables, and a

command area; the other screen (usually the color monitor) showed the application as it executes under the control of the debugger (CodeView).

OS/2 Support

Hardware Requirements: The platform supporting the Dual Display feature under OS/2 is quite specific. The PS/2 line of computers (Micro Channel models 8550 through 8580) includes a VGA display adapter on the system board. To implement dual displays, an 8514/A display adapter or an Image Adapter/A must be installed on the system in addition to the system VGA interface. The following PS/2 displays may be used on the system board connector: 8503, 8506, 8507, and 8508 monochrome displays, and the 8512, 8513, 8514, and 8515 color displays. An 8507 monochrome monitor, 8514 color display or the 8515 color display *must* be attached to the 8514/A or the Image Adapter/A. The 6091 color display must be attached to the Image Adapter with 6091 cable option only.

Software Configuration: The Dual Display feature was introduced with OS/2 Version 1.2. The option for dual display support appears during the initial installation of either OS/2 Standard Edition or Extended Edition. The system actually detects the presence of the 8514/A and asks if there are displays attached to both adapters. If "yes" is the answer, the system unpacks, automatically installs the necessary drivers, and then updates CONFIG.SYS with the proper statements.

If the 8514/A is installed later, it is necessary to manually unpack and copy the correct files to the hard disk. To activate both displays, the CONFIG.SYS file must be changed and the system rebooted. This proce-

cedure is clearly stated in *Operating System/2 Standard Edition Version 1.3 Using Advanced Features* (91F8505) pages 24 through 28, and in *Operating System/2 Extended Edition Version 1.3 User's Guide Volume 1: Base Operation System* (C1F0289) chapter 2, pages 9 through 11.

8514/A Adapter Installation:

Figure 1 shows the procedure to follow after installing the 8514/A Adapter. In the instructions, VGA is the installed device, and the files installed are IBMVGA.DLL and BVHVGA.DLL. The IBMVGA.DLL file was copied to DISPLAY.DLL. To install an 8514/A (with memory, in this case), two additional files – IBMBGA.DLL and BVH8514A.DLL – must be unpacked and copied from the distribution disks to the \OS2DLL subdirectory.

OS/2 Dual Display Definition

Operating System Support: OS/2 allows for 16 simultaneous sessions, or screen groups, to share a PS/2's resources. OS/2 does this intelligently and impartially by allocating time, memory, and access to devices that make up the system. It's important to understand that the PM shell operates in one of these screen groups and can allow OS/2 to manage multiple processes within that shell. This, in effect, shares that single screen group even further. PM uses windows to subdivide the screen group, providing access to the display for various processes.

OS/2 offers limited support to users with dual displays. The 8514/A adapter and the 8514 monitor are reserved for the PM screen group. All other screen groups are directed to the system board VGA and its at-

tached monitor. These include all full-screen applications, OS/2 Full Screen Command sessions, and the DOS compatibility box.

Limitations: When a system has been set up with two displays, and the system is first turned on, the PM screen group appears on the display attached to the 8514/A. The display attached to the VGA port will probably be blank, and remains so until an application requiring a screen group of its own is started. Once a full-screen application has the input focus (keyboard, mouse, and cursor), the PM display becomes static and will not be updated. The applications continue to run. Using ALT-ESC (or CTRL-ESC) causes the system to switch back to the PM display, thereby freezing the full-screen display and allowing the display to be updated.

Only one of the displays can be active at any given time. The inactive monitor continues to display the image (or data) that was on the screen when the focus was removed. That screen cannot be updated until the focus has been returned. The operating system and the PS/2 design do not allow updates to both monitors simultaneously.

CodeView!

What about CodeView? It seems that whenever dual displays are mentioned, CodeView comes to mind and vice versa. This relationship comes from CodeView's history under the DOS and IBM PC environments. CodeView was described earlier as one of the early applications supporting and benefiting from dual displays. Those who've used CodeView on that platform may remember that both displays are updated in realtime.

Find the diskette with the IBMBGA.DLL and BVH8514/A files, insert it into the A: drive, and enter:

```
UNPACK A:IBMBGA.DL@ C:\OS2\DLL
UNPACK A:BVH8514/A.DL@ C:\OS2\DLL
```

Remove the OS/2 disk from drive A: and replace it with the installation diskette. Reboot the system, press ESC at the IBM logo, and enter:

```
COPY C:\OS2\DLL\IBMBGA.DLL C:\OS2\DLL\DISPLAY.DLL
```

Edit the CONFIG.SYS to change the lines that read:

```
DEVINFO=SCR,VGA,C:\OS2\VIOTBL.DCP
SET VIDEO_DEVICES=VIO_IBMVGA
SET VIO_IBMVGA=DEVICE(BVHVGA)
```

to:

```
DEVINFO=SCR,BGA,C:\OS2\VIOTBL.DCP
SET VIDEO_DEVICES=VIO_IBMVGA,VIO_IBM8514A
SET VIO_IBMVGA=DEVICE(BVHVGA)
SET VIO_IBM8514A=DEVICE(BVH8514A)
```

Save the file and close the editor. Turn off the system and place the 8514/A adapter into the slot reserved for it. Turn on the system again and boot from the reference disk to install the .ADF file and to configure the system's CMOS memory. Remove the reference disk and reboot the system again. The system now displays the PM screen on the 8514 monitor and any VIO full-screen program on the VGA display.

Figure 1. Procedure after Installing 8514/A Adapter

When CodeView was used with a single monitor in either DOS or OS/2, the display would "flip" or "swap" as each statement (in single-step mode) was executed. This was done so the output screen could be updated and viewed after each statement. It also slowed the debugging process. CodeView contains an option that turns off screen swapping, letting the debugging process operate faster. If the output screen must be viewed, press F4. Under DOS, you can toggle between the output screen and CodeView. When debugging in OS/2, the F4 key shows the output screen for a (user modifiable) period of time and then returns to the CodeView display. Using two

monitors with DOS and the PC eliminated the need for screen swapping and the time delay encountered while repainting the screen. With OS/2, displays still "flip."

Because of the constraints imposed by OS/2 and the PS/2, CodeView cannot use two displays as it could under DOS on a PC. At best, using two displays allows the programmer to continue seeing the PM desktop while viewing the source code within the debugger. Using both displays applies only if a PM application is being debugged. If a full-screen application is being executed under CodeView, screen flipping is done on the VGA display

with a single monitor in the same way as under DOS.

Further, an application *cannot* use both displays simultaneously. For example, CodeView running under DOS on a PC can control where and when it writes to video RAM, even if the video RAM starts at two different addresses. CodeView can use the monochrome monitor for itself and direct output of the new application under its control to another monitor, which can happen in realtime or dynamically. A loop of code can write a character to video RAM for each iteration of the loop. The character is visible as soon as the character code is placed in memory. This behavior cannot be obtained with OS/2.

Caution! When using CodeView to debug a PM application, it's not unusual to get an overwhelming urge to use ALT-ESC to switch to the PM session to see your application's windows and output. Users commonly forget that they're in a debugging session and switch to PM to access another PM application. *Do not do this!* Doing so locks the system, and the only way out is to turn the system off and on again.

This situation happens while debugging, because at least one instruction has executed. CodeView is trying to exert control over an operating system that, by definition, does not want to be controlled. OS/2 is a multitasking operating system that doesn't allow any application to upset or affect another. CodeView "tricks" OS/2 into letting it control the application being debugged. OS/2 allows this, but in a narrowly defined way. If ALT-ESC is used to switch to PM, CodeView and OS/2 become deadlocked.

The inverse of switching from CodeView to PM is also not al-

lowed. During debugging, the application eventually needs user interaction. At this time, the system focuses on the PM session. (You can check this by moving the mouse. If the mouse pointer on the PM screen moves, it has the focus.) *Do not attempt to switch to CodeView.* If you do, the system locks up, and turning the system off, and then back on is the only way out. The only way to get back to CodeView is to terminate the application or encounter a breakpoint. If the application is being single-stepped and input is required, the focus will remain with the application until the requirement has been satisfied.

With dual displays, users can focus on the PM screen earlier than normal when switching back to the PM session.

Benefits: The advantage of dual displays under OS/2 is the comfort of seeing the Presentation Manager screen group while working in a full-screen session or in the DOS compatibility box. With dual displays, users can focus on the PM screen earlier than normal when switching back to the PM session. The screen update process repaints the various windows while users focus on what they're going to do next. In the case of CodeView and debugging, it is easier to follow the new application code when the application output is constantly visible.

XGA Support

Where does the new XGA Adapter fit into the PS/2 or OS/2 platform?

Support for Extended Graphics Array hardware is the same as for the 8514/A with a few additional features. Anti-aliased text offers easier readability and displayed text has a better appearance. With XGA, there is a choice of resolutions and colors, depending on the amount of display memory available. For a better understanding of the XGA support with OS/2 V.1.3, read pages 27 and 28 in *Operating System/2 Standard Edition Version 1.3 Using Advanced Features* (91F8505).

Summary

PS/2 hardware cannot dynamically update its attached displays in combination with the system board VGA and either the 8514/A adapter or the Image Adapter/A. OS/2, starting with Version 1.2, selects the high-resolution display adapter for the PM session and switches to the planar VGA for all full-screen sessions. An OS/2 application cannot direct text or graphics to a particular display in any way. Because Codeview is a full-screen application, it displays on the VGA-attached display. If CodeView is used to debug another full-screen application, it "screen flips" both CodeView's and the application's output on the VGA display. When debugging a PM application, the "screen flip" still takes place between the two displays. Because OS/2 has to update the PM display, single-step debugging is slow.

ABOUT THE AUTHOR

Larry Pollis is a market support representative in OS/2 Standard Edition technical support. He supports the OS/2 kernel and Presentation Manager. Larry joined IBM in 1988 after eight years with ROLM Corporation, now a subsidiary of IBM and Siemens.

Local Area Networks: The New Utility

*Thomas Toher
IBM Corporation
Research Triangle Park,
North Carolina*

Twenty-five years ago, computers had not yet assumed a place in the office. By the early 1980s, a revolution in computer technology changed how people interact with them.

In the mid 1960s, people interacted with computers by delivering piles of data to keypunch operators who entered them onto punch cards. The trays of cards went to a computer operator who ran the jobs. Checks were issued, reports printed, and records updated at amazing speeds, compared to the manual systems used previously.

Computers were important to all sorts of organizations, but they had not yet assumed a place in the office. Instead, they were locked up in special, environmentally controlled rooms, known as "glass houses," and were tended to by specially trained people who were developing a language of their own to describe their interactions with the computer.

In the 1970s, smaller computers not requiring special environments began to appear in the workplace. The video display terminal and friendlier human interfaces allowed



users to store large amounts of text-based records. The computer became an electronic filing cabinet instead of merely a calculator.

Still, the possibilities for manipulating information stored to meet specific users' needs was extremely limited. By the early '80s, a revolution in computer technology increased densities and decreased prices of storage, and memory dramatically. This changed how people interacted with computers by placing very powerful workstations in the individual's workplace and home.

Today, a personal computer puts more power on your desk than resided in the "glass house" of the 1960s. Yet, this power costs less than one percent of the 1960s-vintage system. More importantly, all that power lets you choose how you want to interact with the computer. Powerful new applications let users store and retrieve information in ways tailored to meet their needs. Now, spreadsheet programs let individuals prepare reports and charts in

minutes. In the 1960s, programmers spent hours creating reports of this complexity. The personal computer, once the office status symbol, has become commonplace.

One of the first things that people noticed as personal computers entered the workplace was that they seldom work in total isolation. Early personal computer users developed their own slow but highly reliable networks for interchanging data and programs. In most organizations it was called "Sneakernet." The floppy disk and its descendants let people walk around the office with an amazing amount of information written on something that weighs only a few ounces. Everyone learned how to use COPYDISK and COPYFILE almost immediately. But Sneakernet, reliable as it was as a data transport mechanism, was too slow and depended too heavily on human relationships. Besides, even the most resourceful user still needed access to information and resources that were best handled by the mainframe.

Personal computers were made to emulate the terminals they recently displaced in the office. When used as an emulator, much of the personal computer's power is unused. Interaction through the mainframe to other PC users was possible, but it was only slightly better than Sneakernet. These users needed a way to share resources in a peer-to-peer relationship. Local area networks (LANs) were designed to meet these new needs.

The requirements for a good LAN are simple. The network must:

- Enhance the power of the devices attached to it
- Be invisible to the user of the attached device
- Have a standard implementation supported by a wide variety of equipment manufacturers
- Be flexible in its physical installation
- Be highly reliable and easily managed

Enhanced Power

LANs enhance the power of the personal computer by giving you access to shared resources such as high-speed printers and data storage devices. The LAN is a high-speed expressway to the information you need, no matter where that information resides. Through bridges and gateways, LANs let you reach out to information anywhere in your organization – to co-workers in the next department, the branch office in Chicago, or the manufacturing plant in England – without ever leaving your desk. Moreover, the LAN is usually faster than dialing a long-distance phone number and making the connection. When the already powerful personal computer is attached to a LAN, all of your organization's resources are at your fingertips.

In addition to sharing the power of other network devices, the LAN lets you share information with other network users. The network pulls your organization together no matter how large or geographically scattered it may be. You can now communicate in moments with colleagues half a world away. And, because the communication is personalized – a note to colleague in England whom you have never met, perhaps – the network helps create new working relationships.

Clearly, the power of the network goes beyond the interactions of machines and the sharing of information and resources. The network can help bind your organization together by its ability to connect people and information.

Invisibility to the User

In a short time, we've become very matter-of-fact about the average person's ability to use computers and networks. Almost everyone uses an automatic teller machine (ATM) occasionally to conduct his or her financial affairs. Using the ATM has become almost intuitive, thanks to careful human factors engineering. If LANs are to succeed in their mission of connecting people to information and to other people, their use must be so intuitive as to be invisible to the user.

Network users must be able to find any information without impairing the access of others and without knowing anything about where information is stored. For example, an insurance claims examiner should need nothing more than the policyholder's name and policy number to gather electronically a complete history of the policyholder's coverage and previous claims. The examiner should not have to know

that the coverage information resides on the mainframe in the San Francisco and the claims information came from five regional claims centers. LANs provide the connectivity to make access to such information possible. Systems analysts and programmers everywhere are working to make that access as simple as your ATM transactions.

Standards and Variety

If the LAN is to be the highway through your organization, it will work only if everyone has access to it, and if all who use it agree upon a set of rules. If LAN access is limited or the rules are unclear, communication is inhibited rather than enhanced.

The computer industry works together to develop LAN standards – rules that permit LANs to operate and to facilitate communications among devices designed by different manufacturers. These standards must be strict enough to ensure that a LAN will function reliably, yet flexible enough so all vendors can provide equipment that can operate as peers on the network. The Institute of Electrical and Electronics Engineers (IEEE) 802 family of LAN standards is widely accepted by both the computer industry and its customers. The predominance of token ring networks and Ethernets is a testimony to how well these committees have done their work. Now the standards are defining how these two networks – and others – should communicate with each other. Many organizations have both token ring and Ethernet networks. For some time, this coexistence was a barrier to building an organization-wide, peer-to-peer network. Now those barriers are coming down as bridges and routers are being developed to link these networks together.

As networks grow larger and new applications require higher data rates, the standards organizations are seeking ways to accommodate these demands with minimal disruption. For example, the IEEE 802.5 Token Ring Network Standard Committee initially developed the standard for a 4 Mbps network. As demand increased, the committee developed a 16 Mbps standard that allowed 4 Mbps token ring users to migrate easily to this more powerful implementation of the token ring standard. Now, the ANSI X3T.9 committee is developing the Fiber Distributed Data Interface (FDDI), a 100 Mbps standard employing a token-passing protocol that should integrate easily into organizations presently using token ring networks.

Physical Flexibility

LANs are being installed in almost every conceivable workplace. Although the industry is developing wireless networks, large-scale implementation of this concept appears to be some years away. For now, LANs depend on cables to link devices in the workplace. Industry standards activities are helping both the computer manufacturer and the user. The Telecommunications Industry of America (TIA) is about to publish a standard for commercial building wiring. That standard, in conjunction with the previously issued one for telecommunications pathways and spaces, provides organizations with advice on designing their buildings to meet telecommunications requirements. By recommending distances from wiring closets to work areas as well as a number of cable choices, these standards provide stable targets for manufacturers developing networks. Unfortunately, the wide variety of data rates and regulatory constraints on electromagnetic inter-

ference (EMI) do not permit the standards to present a single cable choice as a universal media. The development of the standards encourages manufacturers to make their network implementations available on a broad range of the recommended media.

In the contemporary workplace, LANs are being installed as widely as telephones. Manufacturers have made ease-of-configuration (how the network is physically interconnected) a major objective so installation and reconfiguration cause little disruption.

In the contemporary workplace, LANs are being installed as widely as telephones.

Reliability and Manageability

In a remarkably short period of time, LANs have become indispensable to users. Their importance in the workplace demands a level of reliability second-to-none. Many people spend more time using LANs than telephones. The network needs to operate virtually 24 hours a day, seven days a week, without interruption. Further, because LANs have become a utility like the telephone, they should not require a network operator to tend to their needs. Instead, LANs are managed. The IEEE Token Ring Standard provides rules for this management that allow various manufacturers to implement

schemes to meet their customers' needs. IBM's LAN Manager, for example, allows a single personal computer to monitor the activities of large, multiple-segment LANs. LAN Manager also communicates with NetView at the host computer to provide a single point of management for the entire establishment network. With good network management, LANs operate invisibly to serve you continually.

It All Has to Work Together

IBM's commitment to networking extends from the LANs in your office to the wide area networks that connect people around the world. IBM is an active supporter of all industry standards that will help people share information, whether the information is in the next office or on the next continent. LANs are the basic building blocks to this commitment, because they operate in your local environment. If your LAN has the characteristics outlined in this article, it will improve your ability to share information.

ABOUT THE AUTHOR

Thomas Toher is an advisory information developer in IBM's Communications Systems organization. He received a B.A. and M.A. in English from Hobart College and Clark University, respectively. He joined IBM in 1984 and presently works in the local area network information development department. Tom represents IBM on the EIA/TIA TR 41 building cabling, infrastructure, and administration standards group. He is a frequent speaker on cabling and planning for token ring networks at GUIDE and other forums.

And You Thought LANs Were Just for the Office!

Larry Lee
IBM Corporation
Owego, New York

Have you ever thought of a LAN operating in a "harsh environment?" Here's how standard commercial-grade PS/2s operated on a token ring LAN while strapped to a missile launcher on the side of an Apache helicopter. The PS/2s received considerable abuse – from vehicles running over cabling, to dust, rain, vibration and bugs – yet continued to work without a glitch.

One mission of the IBM Federal Sector Division in Owego, New York, is to build "black boxes" for the United States military. We build a variety of mission computers, receivers, and communications equipment, which are used on all kinds of military platforms. As part of this development, PS/2s are used extensively for design, documentation, and testing.

Recently, we had an interesting experience with a semi-automated test system that we developed. We took it to McDonnell Douglas Helicopter Company's facility in Mesa, Arizona, and tested one of these black boxes on an Apache helicopter. This involved using PS/2s to exercise the black box system, collect data, and analyze it in near-realtime. It's not unusual to take a PS/2 into the field to test a system. But this case was unique. We took seven LAN-connected PS/2s and operated them as an integrated system.

The black box system we tested is called the Radar Frequency Interferometer System (RFIS). To the layman, it can be thought of as a big radar detector. We tested RFIS extensively in our anechoic chambers in Owego and were under contract to determine its installed performance on an Apache helicopter.

We planned to install RFIS on the helicopter, radiate it with known radar frequency signals, and monitor the emitter reports that RFIS was generating. We needed to sweep the frequency from a low to a high one in as many discrete steps as possible. At each frequency we collected several samples of two types of data. The information collected at each frequency was a combination of emitter reports on a MIL STD 1553B data bus and other data internal to RFIS. The internal information was accessed through a special test port on the RFIS processor.

After completing one full-frequency sweep, we called on a radio to ask that the helicopter be repositioned to a new azimuth angle. We could then perform another frequency sweep. Because of the manual positioning, it was considered a semi-automated system. We had to wait while the helicopter was manually repositioned on the tarmac in front of our antennas. After we collected data over the azimuth angles of interest, we analyzed the data and produced graphs of the systems' performance. RFIS was tested with the rotor blades turning so we could see exactly what affect the rotating rotor blades had on RFIS system performance.

We performed quite a bit of this testing in our anechoic chambers, but with RFIS mounted on a precision turntable instead of on a helicopter. The chamber data became our performance baseline and was the best our

system could perform because it was operating in a nearly ideal environment. Field test data would point out any degradation in performance due to installation effects of the Apache helicopter.

The test systems used in our chambers were unavailable to us for field work. The systems were part of the anechoic chamber and would be needed for other projects while we were in the field.

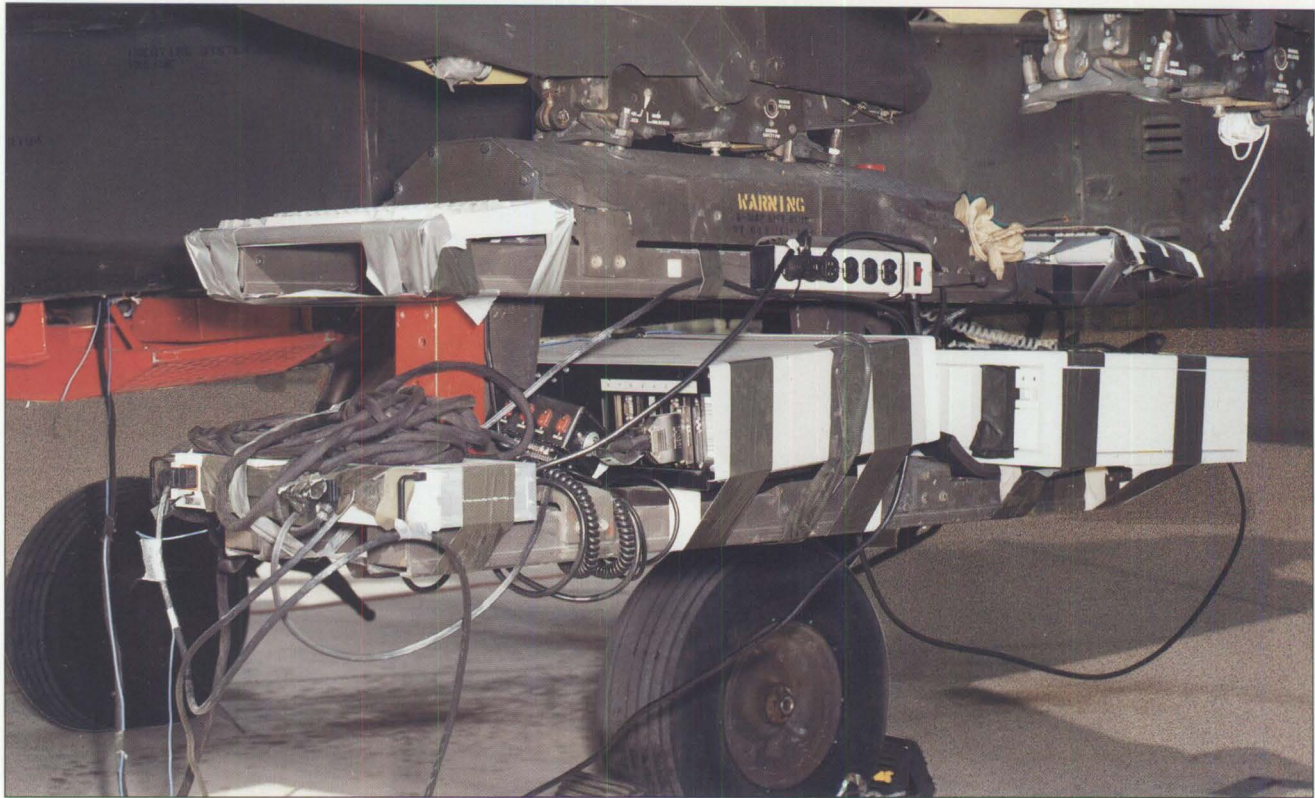
We needed to design an automated data collection system to collect as much data as possible in a limited amount of time.

History

In previous field tests similar to this one, we used a few independently operated PS/2s. This approach worked, but definitely had its limitations. Typically, the RF generators were PS/2-controlled and were located several hundred feet away from the helicopter. There was also a PS/2 located within a few feet of the helicopter to monitor system performance and gather data. The operators at the two sites communicated by radio.

There were constant communication problems between the two sites. This was a primary concern, because we needed radio communication between the sites and the helicopter. Radio problems included rundown batteries and garbled messages caused by noisy helicopter rotors and engines. Only one person could talk at a time. Also, radios are not secure, so we were not always able to freely discuss classified information with each other.

Another concern was personnel staffing. People were frequently out of position during tests, and we did not



want to waste valuable time moving people among test sites.

The unfortunate person who gets the assignment of working under rotating helicopter blades has an especially tough task. An incredible amount of noise is produced by the jet engines and rotor blades on the helicopter, and there is constant buffeting by the rotor wash. Ear protection is a must. It is very difficult trying to communicate and work up to eight hours a day in this noisy environment.

In past tests, we never were able to analyze our data in realtime. Part of the problem was that we did not have instant access to the data being collected. The data from the RFIS system went to a flight recorder on the helicopter and later was transferred to PS/2-compatible format.

The unfortunate person who gets the assignment of working under rotating helicopter blades has an especially tough task.

The transfer process might take a day or two depending on many external factors out of our control.

Analyses would be done overnight after we obtained the data, or over a weekend or after we arrived home. If there was a problem with the data, we might not know about it until two or three days later and maybe

not until the tests were over. To ensure that our data was accurate, we needed near realtime analysis capability to ensure the quality of our data. For these reasons, it was determined that a new data collection system was needed.

The New System

For the new data collection system, we wanted to have instant access to the data. We also wanted to keep our test group away from the helicopter and close together in order to improve communications and minimize transition time. We planned to have a trailer about 300 feet away from the helicopter and most PS/2s and all RF signal generators would be located inside. All the PS/2s would be linked together on the LAN, and the ones outside the trailer would be controlled remotely from inside the trailer. This

approach would allow us to have everyone working in a quiet, clean, safe room. In this room, we could discuss the tests while they were running and see what was going on with any of the PS/2s.

The staff decided on a PS/2-based system on a token ring LAN as the basic system architecture. Because the LAN would be small and temporary, we decided to keep things simple. As a result, we selected DOS-based PS/2s running PC LAN Program Version 1.3, Base Services. The system block diagram is shown in Figure 1.

Hardware Selection

The PS/2s used were not the ones we requested, but we compromised because they were needed quickly. We did not have time to determine how much processor power would be needed to perform any of the tasks or whether the LAN would be fast enough to handle the data rates anticipated. The determination was calculated by the "seat-of-the-pants" method because of our short development schedule. Our "gut feelings" told us we had more "horsepower" than we needed, which turned out to be correct.

Equipment

Three PS/2 Model P70s were used. We were unfamiliar with the P70s, but after using them in the lab for a few days, they became prized possessions because they performed well.

The PC AT was used, because it was an existing piece of lab equipment with a special-purpose card and software. We added a LAN card and upgraded DOS and added PC LAN Program software.

The two PS/2 Model 80s (071 and 321) and a PS/2 Model 70 used

were normal desktop units. They were not industrial-grade or ruggedized.

The LAN cards were the 16/4 megabit variety except for the one in the PC AT. This was one of the older 4 MB cards, which resulted in the LAN running at the slower 4 MB rate. But the slower data rate never posed a problem.

The Multistation Access Units (MAUs) used were standard, non-ruggedized, rack-mounted 8228 units.



Throughout our time in Arizona, the STS and BUSMON PS/2s were attached with duct tape to a missile launcher, which was on the side of the helicopter.

After we had our basic system architecture in place, there were still a lot of questions about exactly where equipment would be located and how things would be interconnected. Some of the decisions about where and how PS/2s would be mounted were not made until we got to Mesa and showed the McDonnell Douglas staff our equipment and the lengths of some of our cables.

The connections from the Software Test Station (STS) and Bus Monitor (BUSMON) PS/2s to the RFIS system proved to be quite difficult. Removing our test cables from the RFIS system involved lowering an instrumentation pallet from the bot-

tom of the helicopter, which took about a half hour, 30 seconds to break the connections, and another half hour to put everything back together. For this reason, we left the STS and BUSMON PS/2s permanently connected to the RFIS system. Throughout our time in Arizona, the STS and BUSMON PS/2s were attached with duct tape to a missile launcher, which was on the side of the helicopter.

The mounting scheme is shown in the photo on the second page of this article. This approach was certainly less than ideal, but was the best we could do in the limited time available. No vibration analysis on the way the PS/2s were mounted was ever performed. If the PS/2s broke, we knew we could call IBM Service and have them repaired or replaced overnight. If one broke, we couldn't let the helicopter sit idle while we waited to get our test equipment fixed.

Remote Control

Because the STS and BUSMON PS/2s were mounted on the missile launcher, neither of these PS/2s could have displays attached. Therefore, these PS/2s were operated exclusively by remote control. Fortunately, there is good variety of remote control programs available for use on LANs, and we selected one called "PCNETWRK." The program was easy to install and use. With just a few small batch programs, we set it up so any PS/2 in the network could control and reboot any other PS/2. With this program, we often had three PS/2s in the trailer simultaneously controlling three other PS/2s outside the trailer with no performance degradation to the LAN.

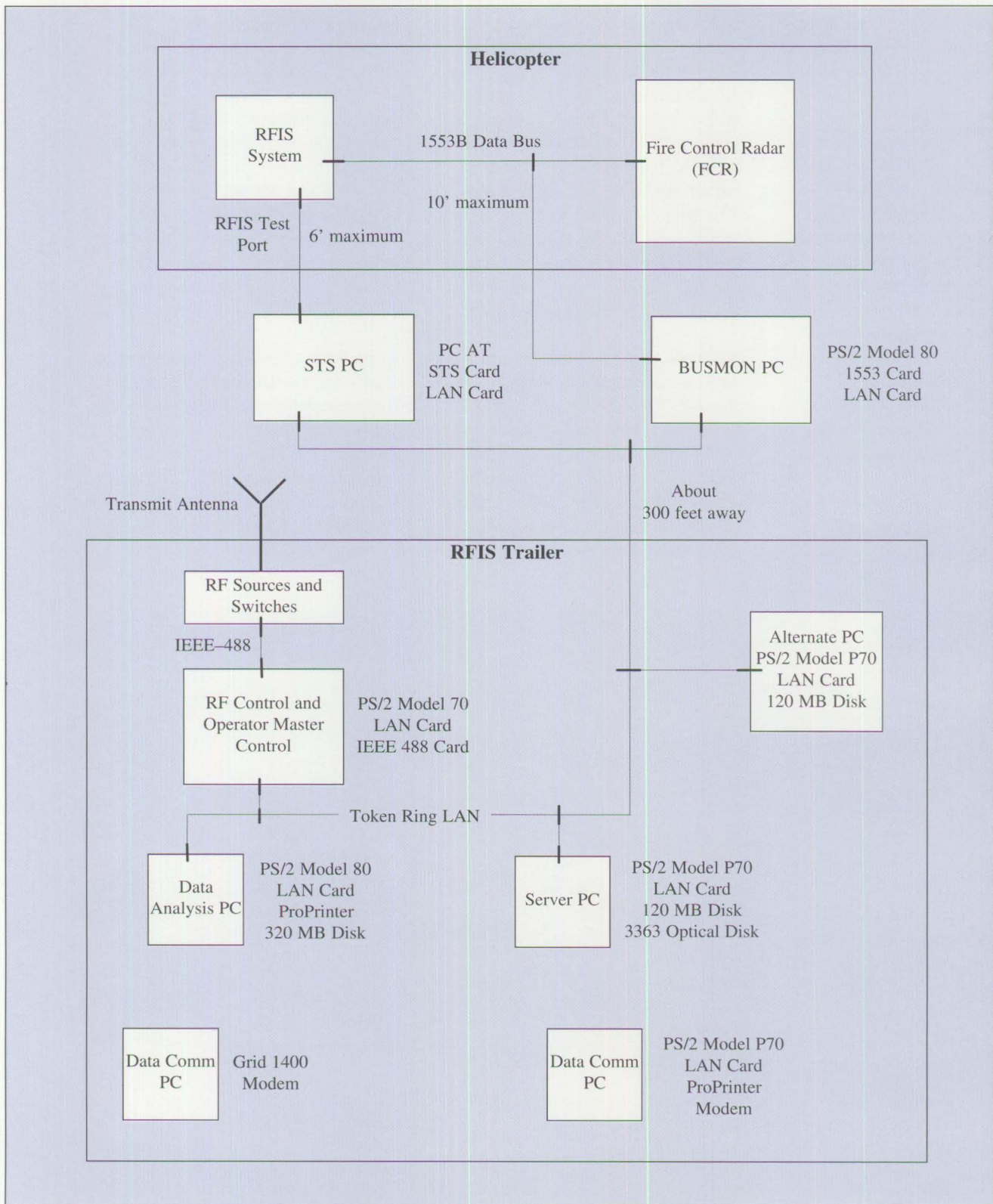


Figure 1. RFIS Longbow Characterization Field Test

The STS was placed under remote control first thing in the morning and stayed that way until we powered off the helicopter in the evening. The BUSMON PS/2 was under remote control only when we started it or if a problem was suspected somewhere. The RF CONTROL PS/2 was originally in the trailer, and it was the PS/2 that controlled the sequence. Later in the series, this PS/2 was placed on a cart and moved outside the trailer (see photo on this page) and was controlled remotely. Using remote control versus a display proved advantageous, because we found it was nearly impossible to read the display in the bright Arizona sun.

Hello Mesa!

During the first days of testing, we were fine-tuning our system while

collecting data. After collecting data for only a short time, we changed some of the BUSMON PS/2s operating parameters to speed up our data taking. Using the remote control program, we had the BUSMON PS/2 recompile its program with the new parameters in the middle of a test. This happened while the helicopter blades were turning. The only thing the people in the helicopter noticed was that one data point took slightly more time than the previous ones.

After collecting data for several days, we observed that the BUSMON PS/2 hung up several times daily. Upon noticing this, we switched to remote control and all we saw was the DOS prompt. The program was "blown" for no apparent reason. However, we rebooted, and it functioned properly. After a

few days, we looked at the BUSMON code. We found that pressing any key cleared the screen and returned to DOS. We guessed that a key on the Model 80 was being hit by something blown by the rotor wash. After an unsuccessful trial with a cardboard cover on the keyboard, we found the solution. Taping the spacebar so it couldn't activate prevented any further problems. Excessive vibration activated the spacebar.

At the end of the day, shutdown procedure for the helicopter-mounted PS/2s consisted of turning the power off, unplugging the extension cord, detaching the LAN cable from the helicopter-mounted MAU, and cleaning the dead bugs and dirt from the PS/2s. The LAN cable was rolled up



and stuffed under the trailer. We had no protective covers for the PS/2s.

During the first days of testing, the helicopter was pulled into a hangar at night for maintenance. Later on, it was left outside all night with the PS/2s still attached. Fortunately, there was no dew in the morning. By testing in the autumn, we didn't have to worry about excessively high temperatures. We never had a temperature-related failure of any kind.

Cables

We used standard IBM LAN cables intended for use indoors. The cables in the trailer were on the floor and taped down for safety. The 300-foot cable from the trailer to the helicopter consisted of two 150-foot cables with the connectors taped together. This cable was rolled up every night and stored under the trailer. In the morning, the cable was rolled out

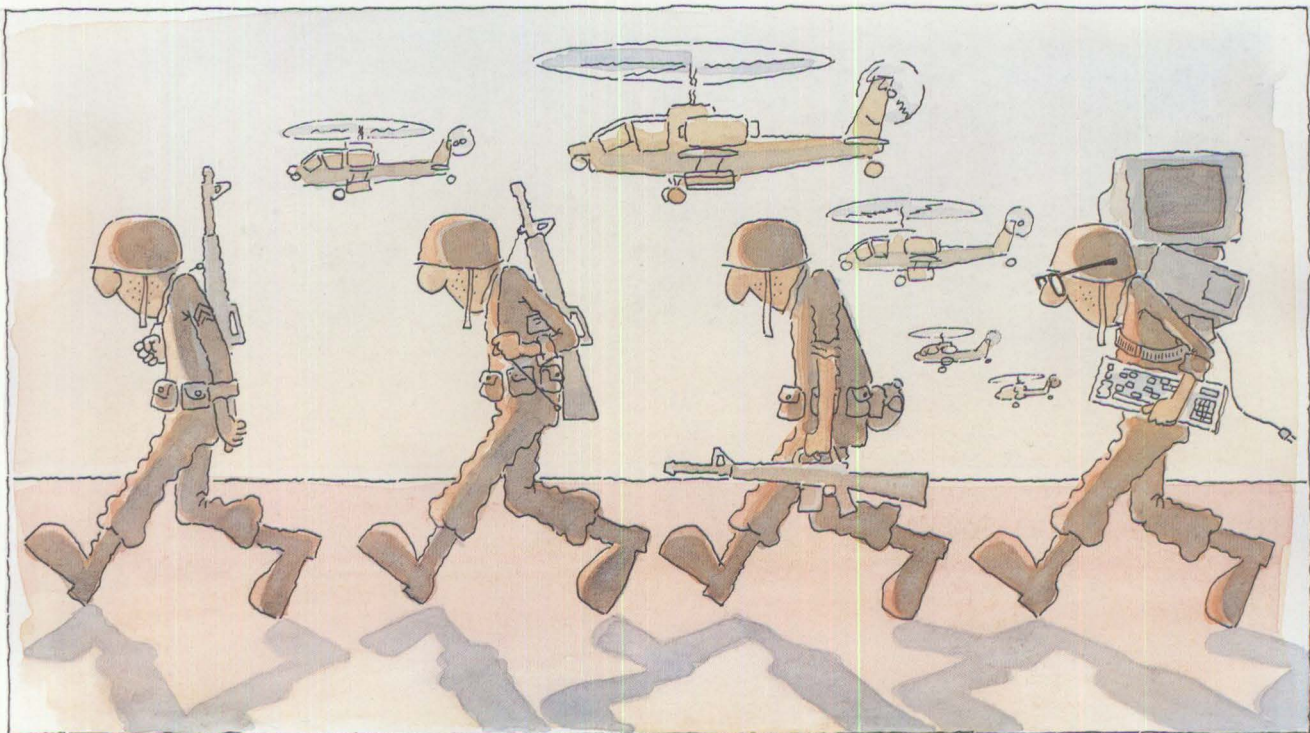
and connected to the MAU on the helicopter and secured with tape.

The cable near the helicopter was secured with six to ten "shot bags." These were cloth bags about the size of a five-pound bag of flour filled with lead shot. Each weighed about 30 pounds. The weight prevented the cables from being picked up by the rotor wash and sucked into the rotors or engines.

The cable was run over several times – the first time by a fire truck, which put noticeable flat spots on the cable but didn't affect data transmission.

The only special precautions for dust or water were taping shut the floppy disk drives on the two helicopter-mounted PS/2s. We didn't tape the air vents on either of those PS/2s. It rained briefly one night, but fortunately the helicopter was in the hangar. The cables were laid on a wet tarmac, and when the helicopter was rolled out, everything worked.

The cable was run over several times – the first time by a fire truck, which put noticeable flat spots on the cable but didn't affect data transmission. A few cars, small tractors, and the helicopter's tail wheel also ran over the cable without any noticeable damage. However, when a small tractor pulled a big steel fixture over the cable, its steel wheels put very noticeable flat spots on the cable. Again, data integrity on the LAN was unaffected.



Picking Up Good Vibrations...


After 10 days, we changed the setup to perform a different series of tests. We took the RF-controlled PS/2 and the RF generators out the trailer and moved them closer to the helicopter. Everything seemed to run fine for a while, then we noticed an occasional anomaly in the data – apparently there was a problem with the BUSMON PS/2. We tried to fix the problem remotely with a software patch, but it didn't work. Figuring we might have an intermittent connection in this PS/2, I monitored it while the rotors were turning. Standing under the rotors, I was astounded by the incredible vibration and noise to which the PS/2s were subjected. The vibration was much worse outside on the missile launcher versus inside the helicopter.

We shut down the rotors, took the cover off the Model 80, yet found no internal problems. All cards and cables were still in place. Later, we discovered a software timing problem on the LAN caused by the way the BUSMON software was written. After reconfiguring, we discovered that there was too much delay in its one critical path. After a small configuration change, everything was working again. The Model 80 was subjected to tremendous vibration all day long, yet continued to work flawlessly.

The two PS/2s on the helicopter were being subjected to incredible vibration, which was totally unexpected. We knew there would be some vibration, but not nearly as much as we saw. Had we anticipated this, we never would have operated in this fashion. We continued using

the mounting scheme hoping that because both PS/2s had made it this long, they would last a few more days. Both PS/2s still worked perfectly when we got back to Owego, despite 15 days of torture.

One thing that probably helped our disk drives survive this vibration was a head-parking terminate-and-stay-resident (TSR) program. This program, which parks disk drive heads after 15 seconds of inactivity, was installed on all PS/2s.



The testing was successful because of the flexibility and performance of the token ring LAN, augmented by the amazing reliability of the PS/2s.

Other Things

We had two modem-equipped PS/2s that allowed us to access the Owego VM systems. We used these links to transfer preliminary results to other engineers in Owego, and to obtain their suggestions, and to download software.

As shown in Figure 1, the Grid laptop PS/2 remained in the trailer. The modem-equipped PS/2 Model P70 was taken to the hotel nightly to prepare the next day's test card, upload and download data, and to develop programs.

Conclusion

Despite a host of problems – excessive vibration, dust, temperature variation, and unusual abuse of the LAN cable – we completed the entire 15-day test sequence with no PS/2 or LAN hardware failures of any kind. We collected over 60 million bytes of data from RFIS, far more than any previous test session.

We couldn't do as much near-realtime data analysis as planned, because the scope of the task was much greater than originally estimated. However, we did sufficient analysis in Mesa to ensure the accuracy of the data. We caught several problems early on, allowing us to recover and successfully complete the tests. Final analysis of the data required two months' work.

With the LAN system used in this test, we were able to detect problems with RFIS, with our own test equipment, and with equipment on the Apache. In fact, the testing was successful because of the flexibility and performance of the token ring LAN, augmented by the amazing reliability of the PS/2s.

ABOUT THE AUTHOR

Larry Lee is a staff engineer at IBM's Federal Sector Division in Owego, New York. Presently, he is working as a systems engineer on the AN/ALR-76 ESM system used on the S3B aircraft. Since joining IBM in 1977, Larry has worked on a variety of electronic support measures systems for military customers. He received a bachelor of technology degree from the State University of New York at Binghamton.

Remote LAN Management Tools

Daniel J. Mieczkowski and
Marvin W. Boswell
IBM Corporation
Research Triangle Park,
North Carolina

The wall chart "IBM LAN-Based Communications Protocols" is inserted in this issue and accompanies this article.

The IBM 16/4 Token-Ring Trace and Performance Program (TAP) and the IBM Distributed Console Access Facility (DCAF) constitute a powerful LAN management tool when used in conjunction with one another. This article details the features of each of these programs and describes the advantages of using them together.

In today's environment, when a workstation on a local area network (LAN) encounters a problem, a technician must physically go to the site to determine what caused the problem. On-site, physical interaction is time-consuming and defeats the purpose of effective distributed processing.

With the IBM Distributed Console Access Facility (part number 73F6159), operators can take control of a DOS or OS/2 workstation and troubleshoot remotely. Operators from the controlling workstation can now manipulate or monitor OS/2 full-screen applications or DOS text applications on the target workstation (Figure 1).

One tool available for tracing LAN traffic and monitoring LAN performance is IBM's 16/4 Token-Ring

Network Trace and Performance (TAP) program. When used in conjunction with DCAF, it provides remote problem-determination capabilities previously available only on site at the local ring.

Distributed Console Access Facility

Situation 1: Imagine this situation. Someone in the network is having a problem with a PS/2 application. Finally, after becoming totally frustrated, the user calls the help desk. After a lengthy discussion, the help-desk operator reaches an impasse.

To troubleshoot effectively, the help-desk operator feels the only solution is to see what the user is doing.

Situation 2: A large distributed network with multiple LAN servers is becoming an administrative nightmare. Each location requires trained personnel to handle day-to-day LAN administration tasks that could be handled more effectively from a central location.

The Solution: There is a solution for both the user and the help-desk operator – IBM's Distributed Console Access Facility. With DCAF, the operator can take complete con-

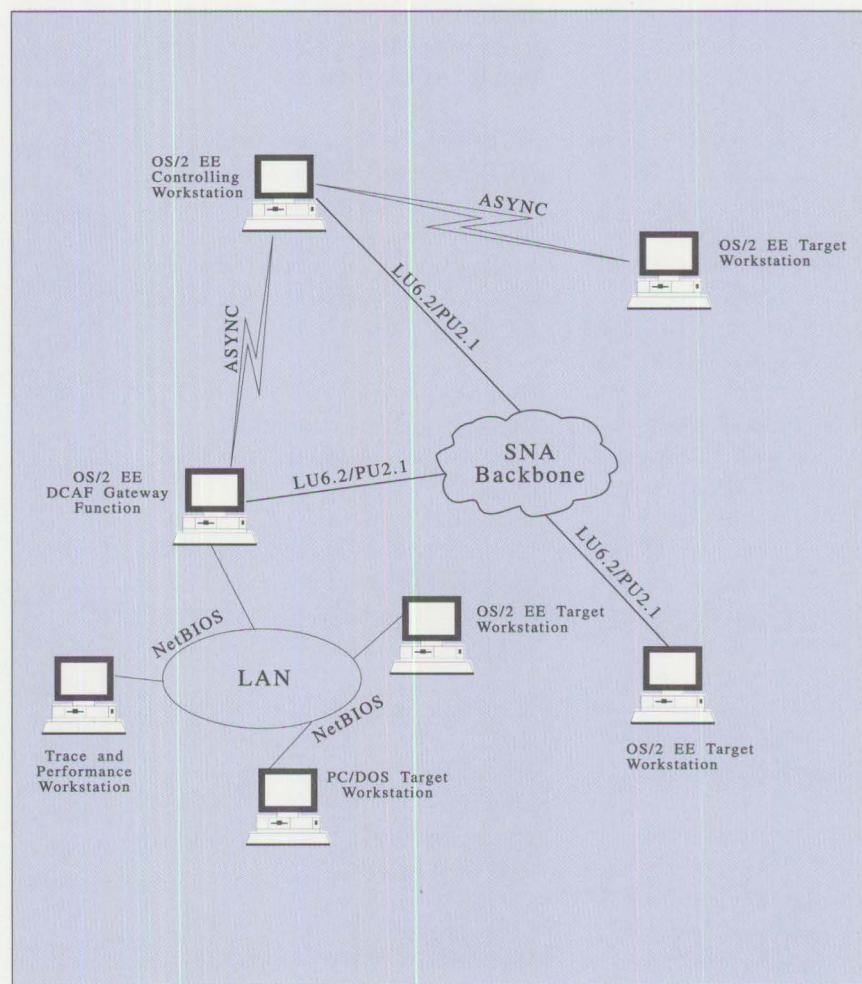


Figure 1. Distributed Console Access Facility (DCAF)

trol of a remote workstation or monitor and see possible errors the user is making. The operator can take over the workstation as if sitting in front of it.

From a central location, you can remotely monitor or control DOS or OS/2 workstations. Communication can be through your existing SNA™ network, a local area network, or through an asynchronous connection.

Once you have taken control of a workstation, you can perform functions such as configuring Communications Manager profiles, viewing error logs, performing LAN server administration tasks, or running the trace and performance program.

Terminology: A DCAF environment is comprised of different components. The controlling workstation is the one that will initiate all DCAF sessions and takes over another workstation, known as the target workstation. The controlling station requires OS/2 EE 1.2 or higher, while the target station can be either OS/2 or PC-DOS 3.30 or higher.

When a controlling workstation establishes a session with the target workstation, the session will be in one of two modes – active or monitor. In active mode, the controlling workstation operator takes over both the keyboard and display of the target workstation. In monitor mode, the target workstation operator retains control of the keyboard, but the display can be viewed at both controlling and target workstations.

The DCAF gateway component is used to access LAN targets. The DCAF controlling station will establish a session with the DCAF gateway, which acts as a single entry point into a LAN. This configuration is required to establish a DCAF ses-

sion to a DOS target, but is also advantageous when accessing OS/2 targets on a LAN. When the DCAF gateway is used, configuration requirements are reduced, and the need to run Communications Manager on the targets is eliminated.

When using the DCAF gateway, you also need the DCAF LAN directory component. This component maintains a list of available targets and their status. When a connection is made between the DCAF controlling station and the DCAF gateway, this list is displayed on the controlling station, allowing the operator to select the target for the DCAF session.

Communications: The following types of communications are found in a DCAF environment:

Switched asynchronous – uses the ACDI function of Communications Manager. This method can be used for communications between the controlling station and an OS/2 target, or between a controlling station and the DCAF gateway

SNA – uses the APPC function of Communications Manager over all OS/2-supported Data Link Control protocols (for example, IBMTRNET, SDLC, X.25, and so forth). This method can be used between a controlling station and an OS/2 target, or between a controlling station and the DCAF gateway

NetBIOS – is the protocol used between the DCAF gateway and targets on the LAN (token ring, Ethernet, or PC Network). This is the only method of communicating with DOS targets.

Environment: A limitation of DCAF is its inability to work with Presentation Manager screens. This, however, does not mean that only

workstations with full-screen text sessions can use DCAF. One of its capabilities is to open a full-screen session on an OS/2 target. Although the limitation still exists, the controlling DCAF operator can access all full-screen sessions of the workstation.

One DCAF function available exclusively on OS/2 workstations is file transfer capability, which is a bidirectional, single file function. To keep DOS memory requirements as low as possible, this function was not included for DOS targets. To use DCAF for transferring DOS files, first copy the file to an OS/2 server that is also a DCAF target. Then establish a session with the OS/2 target and use the DCAF file transfer function.

Trace and Performance

The IBM 16/4 Token-Ring Network Trace and Performance (TAP) tool lets users trace and analyze 16 or 4 Mbps token ring networks, as well as monitor performance of those rings. TAP is used for problem isolation and determination, optimizing ring performance, and as a token ring application development aid. This tool operates in three modes:

- Trace
- Performance monitoring
- Traffic matrix

The TAP program, part number 93X5688, runs under DOS 3.30 or higher, and requires an IBM 16/4 Trace and Performance adapter, part number 74F5121 (AT Bus), or part number 74F5130 (Micro Channel). When not used with the TAP program, they will operate as normal token ring adapters.

When the adapter is used with TAP, the adapter transmits a sequence of

“trace present” frames to the LAN Manager functional address to announce its presence on the token ring. If a LAN Manager anywhere on the network is configured to prevent tracing, the LAN Manager automatically issues a “remove ring station” frame. This causes the TAP adapter to remove itself from the ring, which inhibits any tracing. This security function is built into the TAP adapter and cannot be disabled.

Trace: In trace mode, the TAP tool copies all frames on the ring segment regardless of the destination address. Various trace options allow you to trace all or selected data, such as:

- Data to trace
 - All frames
 - Media Access Control (MAC) frames only
 - Non-MAC frames only
- First buffer or entire frame
- Addresses
 - Destination
 - Source
 - Destination *and* source

Up to 11 MB of data may be captured using multiple data files.

Use the trace analysis facility (RTAP) to review and analyze the data captured by the trace facility. RTAP enables you to display, print, or store the ring network address configuration, which is a summary of the frames captured by the trace facility, and the contents of each frame.

The trace analysis facility provides protocol interpretation capability to help you understand the frame data. Interpretation is provided for the following protocols:

- MAC IEEE 802.5 (token ring)
- Logical Link Control (LLC) 802.2
- SNA
- NetBIOS
- TCP/IP

These protocols and the interrelationship between them are illustrated in the IBM LAN-Based Communications Protocols chart included as an insert with this issue. Contact your IBM representative for information on classes or manuals that clarify the information in this chart.

Performance: The performance facility is used to obtain token ring network performance and utilization measurements. This facility counts the number of frames and bytes passing through the ring on which the TAP adapter is installed and saves this information in a file. By analyzing the performance data, you can obtain throughput measurements and determine ring utilization.

While the performance facility is operating, it displays either of two panels that show the realtime performance of the ring in different formats. The first panel (Figure 2) displays two numbers and two bar graphs reflecting ring utilization. The upper number and bar graph represent bandwidth utilization of all frames on the ring. The lower number and bar graph represent the utilization of user data contained in the frames on the ring. The difference between the two values can be interpreted as the MAC and LLC protocol overhead. This display is updated every two seconds.

The second panel uses a bar graph to illustrate the ring utilization over last two hours. This graph is updated every four minutes.

From the data collected by the performance facility, the analysis facility creates a summary of the overall ring utilization. The summary includes:

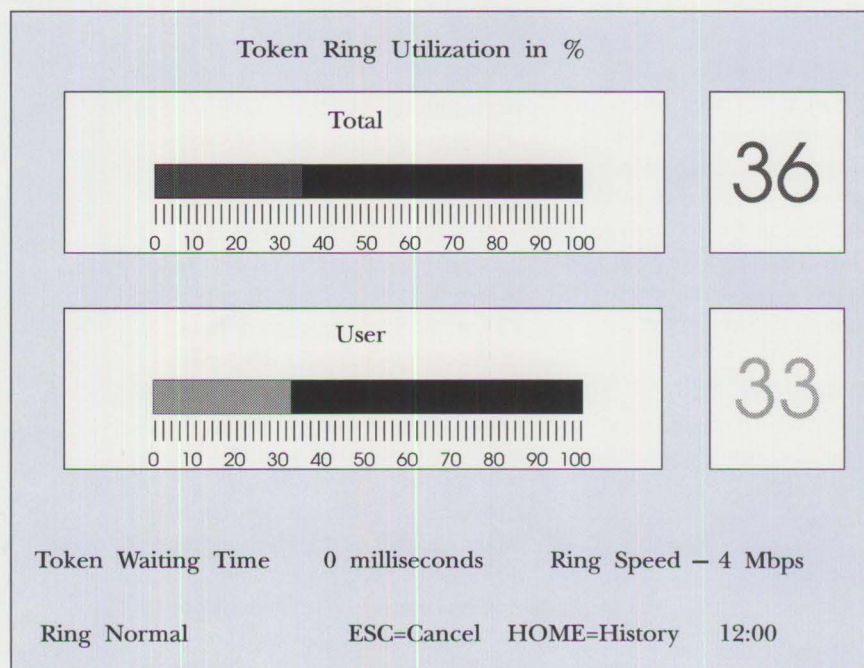


Figure 2. Token Ring Utilization in % Panel

- Time period
- Total number of frames per second and bits per second
- Average percentage of bandwidth utilization
- Frame size distribution percentages

The analysis facility also creates graphs illustrating average performance measurements and tables showing the amount of ring traffic over a period of time.

Traffic Matrix: The traffic matrix facility collects information about non-MAC frames sent between station pairs on the ring. This information can be used to determine:

- Which stations are most active
- How many frames each station is sending and receiving
- Type and size of these frames

With this facility, you can also determine whether particular station pairs are exchanging unusually large amounts of data.

Uses for TAP: There are many ways TAP is useful. Token ring communication problems are determined by using the trace and analysis portions of the tool. Performance mode and traffic matrix mode are used to determine how to reconfigure the ring for improved performance, throughput, and response time.

TAP Through DCAF

Both DCAF and TAP can be used for LAN management. They work together to provide enhanced LAN management capabilities. All TAP program functions operate the same under control of DCAF as they do

when operated at the local TAP workstation.

TAP traces frames or measures utilization only on the ring segment to which it is connected. To provide the best coverage for your network, TAP workstations should be placed on the most heavily used rings in the network. This should be done unless you plan to install a TAP workstation on every ring segment. For example, you may want TAP workstations on all backbone or apex rings or on any ring with a 3174 or 37X5 gateway. You may also want to consider a portable PS/2 that can be moved from ring to ring when necessary. Regardless of the number and placement of the TAP workstations, they should all be DCAF-accessible. This allows tracing and performance measurement of remote rings from a central location.

TAP/DCAF Requirements: In addition to the DCAF and TAP system requirements, there are two other requirements for using TAP through DCAF. First, there must be two token ring adapter cards in the TAP workstation – one 16/4 TAP adapter and one token ring adapter. Second, Version 2.01 of the IBM 16/4 Token-Ring Network Trace and Performance program is required. This version is available at no charge to owners of Version 2.0. Simply order PTF #UR33813 through normal IBM service channels. This PTF changes the way the adapters are initialized by the TAP program, which allows it to be used with DCAF.

Procedure: DCAF should be included as part of the startup procedure on the DOS workstation that will be running TAP. The batch file DCAFDOS.BAT is created during DCAF installation. By including this file in the AUTOEXEC.BAT, the

workstation will be started as a DCAF target.

There are two adapters in the TAP workstation – one will be used for DCAF, and the other for TAP. At installation, DCAF will be configured to use either the primary or alternate adapter. Every TAP operation allows you to select the adapter to use with a default of *primary*. The DCAF session will terminate if TAP attempts to use the same adapter as DCAF. Ensure that TAP and DCAF use different adapters.

Using parameter files with TAP helps users select the proper adapter to be used with DCAF. Parameter files provide a convenient way to configure various setups required in the TAP program, as they need to be selected and entered only once. On subsequent uses of TAP, the file can be recalled by entering the name on the command line.

Remote LAN Management

Delays seem to be an inherent part of current problem determination procedures. A user calls the help desk to report a problem on a LAN-attached workstation. The help-desk operator cannot resolve the problem, so it is routed to the next level of support. A network technician responds to the call and determines that a trace is needed. After locating the TAP workstation, a trace is completed, yet the technician needs help in analyzing the information. The trace file is copied (or printed) and passed to someone else for analysis. Eventually, the problem is resolved, but considerable time has passed. And usually, this time is significantly increased when dealing with remote locations.

We now have the remote LAN management tools that will greatly re-



duce these delays. DCAF can be used by the help-desk operator, the network technician, as well as others involved in problem determination and resolution. If DCAF is installed on all workstations on the network, the help-desk operator can see the problem more clearly, which results in faster problem identification.

The network technician can use DCAF to trace the problem without typical delays. Even when the technician needs assistance in analyzing

the trace, DCAF can help. Either the trace file can be copied to an OS/2 server and transferred using DCAF, or someone else can establish a DCAF session with the TAP workstation and view the trace file online.

There are other advantages for using DCAF and TAP for remote LAN management. Remote ring performance can be monitored, allowing problems to be anticipated and changes made before they actually occur. More problems can be man-

aged and resolved from a central location by taking advantage of local expertise. This can also reduce training requirements at the remote locations. The benefits of using these products include reduced time and expense in managing the network and more efficient use of LAN expertise.

Using the Tools

Let's look at some situations where these products prove useful.

Situation 1: Your organization uses 3174s as gateways for 3270 emulation. This weekend you are adding several new 3174s to the network. All of the workstations using these new control units as gateways need to have the destination address changed in the Communications Manager profile. There are two ways to approach this task – you can search out every workstation, or you can use DCAF to make the changes from a central location. The first method is laborious and time-consuming; the second is efficient and fast.

Instead of searching from floor to floor and office to office to locate the workstations that need changing, you can remain in one location and access the stations with DCAF. With DCAF, Communications Manager profiles can be displayed, created, changed, or deleted. All of these changes can be made in a fraction of the time it used to take.

Situation 2: A help-desk operator receives a frantic call from a user – everything worked fine yesterday, but today the user can't sign onto the host. The screen shows a COM695 error. The user insists that nothing has been changed, but for some reason, the system is not working. The help-desk operator opens a problem report and assigns it to a network technician.

The network technician receives the problem report from the help desk and begins problem determination. Because the problem seems to affect only one user, the technician suspects that perhaps something really *has* been changed. Experience has shown the technician that the fastest way to resolve this type of problem is with a trace.

This network is designed with a TAP workstation with DCAF on all rings with 37X5 gateways. The technician starts a trace on the appropriate ring, selectively tracing only the workstation and gateway addresses. The technician calls the user so the operation can be tried again. Of course, the connection still fails, and the COM695 error is still appearing, but now a trace is running, and the technician will soon see exactly what is happening.

DCAF adds a new dimension – electronic monitoring from anywhere on the network.

The technician analyzes the trace by comparing the sequence of frames with an existing trace of a successful connection. A mismatch is found, indicating the workstation is sending some incorrect values from the Communications Manager profile.

Without leaving the help desk, the technician establishes a DCAF session with the user's workstation, changes the incorrect value in the Communications Manager profile, contacts the user, and asks the user to retry the operation. This time it works! The problem has been resolved by taking advantage of the tools available for remote LAN management.

Conclusion

Before the release of the Distributed Console Access Facility and the current release of the 16/4 Trace and

Performance Program, technicians had to physically monitor the site experiencing difficulty. DCAF, however, adds a new dimension – electronic monitoring from anywhere on the network. The combination of DCAF and TAP makes best use of network experts by allowing them to remotely resolve users' system problems. These tools are invaluable to those responsible for managing and maintaining a local area network.

ABOUT THE AUTHORS

Daniel J. Mieczkowski is a staff programmer in IBM's Communications Systems organization. Dan is licensed as a registered professional engineer. He joined IBM in 1989 and presently works in the LAN Microcode department. His present responsibilities include programming on the 8230 Maintenance Facility and the 16/4 Trace and Performance Program. Dan received a B.S.E.E. from the University of Pittsburgh and M.S. degrees in engineering management and computer science from the Florida Institute of Technology.

Marvin W. Boswell is a staff programmer in IBM's Communications Systems organization. He works in the programming laboratory as a team leader in DCAF development. After joining IBM in 1984 in Endicott, New York, his assignments have included PC/VM Bond, the Enhanced Connectivity Facility, the Shared File System for CMS, REXX for OS/2, and NetView/PC API Gateway. Marvin received a B.S. in computer science from the University of Virginia.

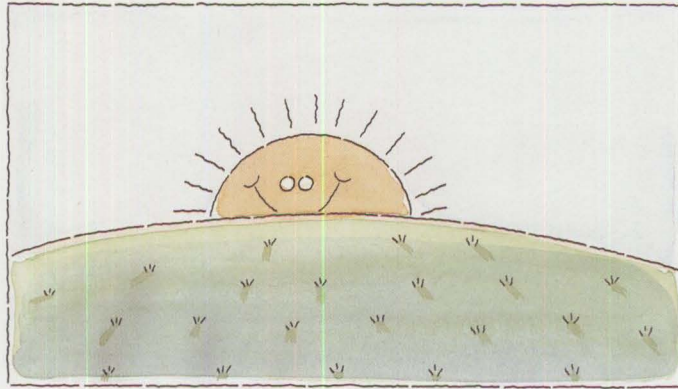
New Horizons for IBM's Shielded Twisted-Pair Cabling

*Hank Foglia and Thomas Toher
IBM Corporation
Research Triangle Park
North Carolina*

Today's appetite for information and expanding applications has placed enormous requirements on the network transport system. This article explores solutions to these increasing demands that allow transmitting high-speed data and full-motion analog video on standard shielded twisted-pair cabling.

In 1984, when the IBM Cabling System first introduced its 150-ohm, shielded twisted-pair (STP) cable, the typical computer user communicated through a terminal to a computer in a "glass house." Today, computing power that only a few years ago would have cost tens of thousands of dollars is sitting on desktops all over the world. This article focuses on the latest advances in transmitting 100 Mbps signals on STP as well as the simultaneous transmission of broadband video and baseband signals on STP.

Today's workstation still needs to communicate with a host computer to access common information and occasionally to take advantage of the mainframe's massive processing power. In 1984, the first devices to use IBM's new cable communicated with host systems at data rates that did not fully exploit the capabilities of the cable.



Even the 1985 introduction of the 4 Mbps IBM Token-Ring Network did not begin to stretch the cable's inherent capabilities. The 16 Mbps implementation of token ring seemed for a while to define the comfortable limits of the cable's utility. Now, the experimental work of IBM engineers at Research Triangle Park has shown that there is life – and bandwidth – in the IBM Cabling System 150-ohm STP cables.

IBM's STP cables can support high data rate transmissions because of the shield that protects the environment from electromagnetic interference (EMI). Equally important is that the signal on the cable is protected from noise in the environment surrounding the cable. Furthermore, the shield structure helps to maintain the cable's physical and transmission integrity.

The fundamental issues in telecommunications transmission are easy to express. First, cable utility is limited by the laws of physics; and second, transmissions are regulated by the laws of man, notably regulatory requirements governing radiation of electronic signals into the environment. These two sets of laws often interact in ways that are challenging to transmission systems engineers. In other words, not everything possi-

ble in the realm of physical law is permissible by regulatory law.

IBM engineers have had to look for ways to expand the utility of IBM Cabling System cables that stay within the bounds of both sets of law. Further, cables must still deliver the level of reliability that IBM Cabling System users have come to expect.

High-Speed (100 Mbps) Transmission on STP

The American National Standard Institute (ANSI) Fiber Distributed Data Interface (FDDI) Standard Committee is currently investigating transmitting at 100 Mbps on copper lobes. IBM and other vendors have made a series of proposals to the standards committee. The IBM proposal addresses only 150-ohm shielded twisted-pair cabling (equivalent to IBM's type 1 cable) as described in the Electronic Industries Association (EIA) Commercial Building Wiring Standard.

IBM's experience with typical unshielded twisted-pair indicates that it is not a viable alternative for high data rate (100 Mbps) applications. This is because the high levels of electromagnetic interference (EMI) generated by such signals are difficult to reduce to acceptable limits in a design that is both technically

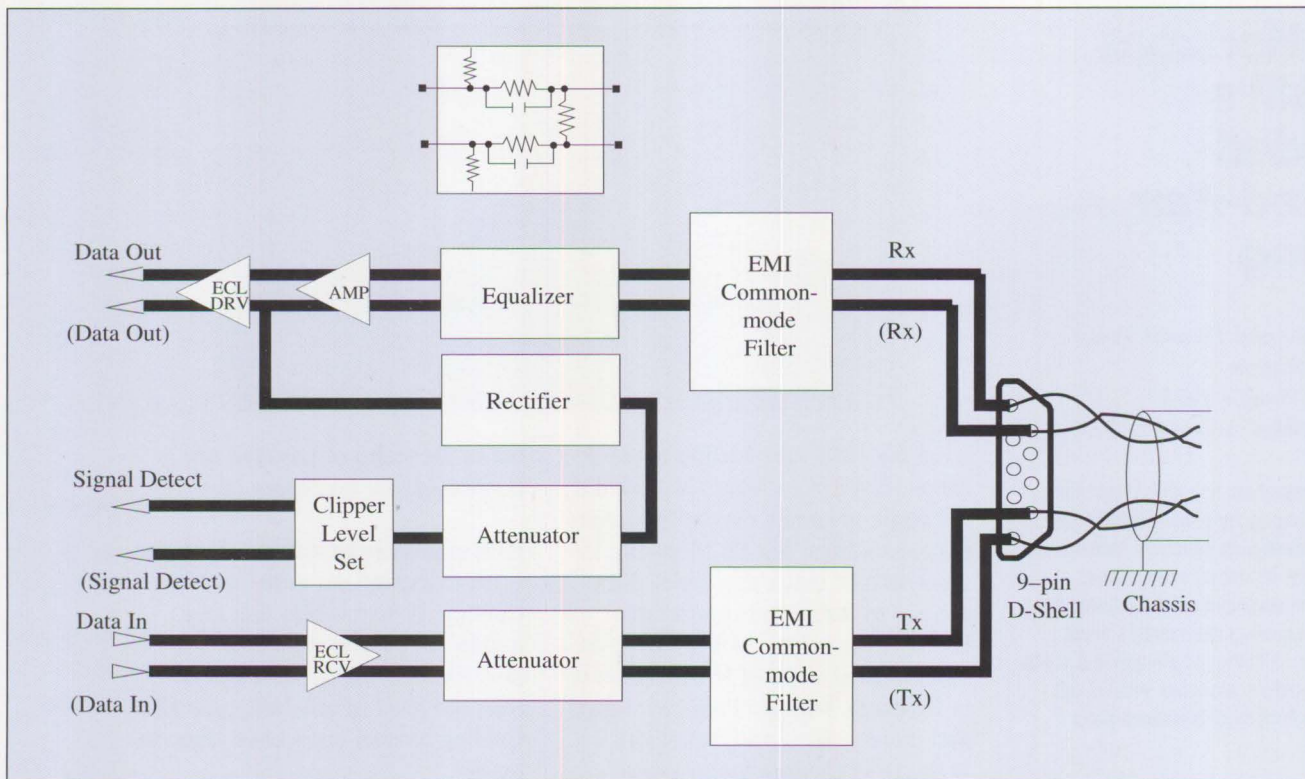


Figure 1. Transceiver Block Diagram

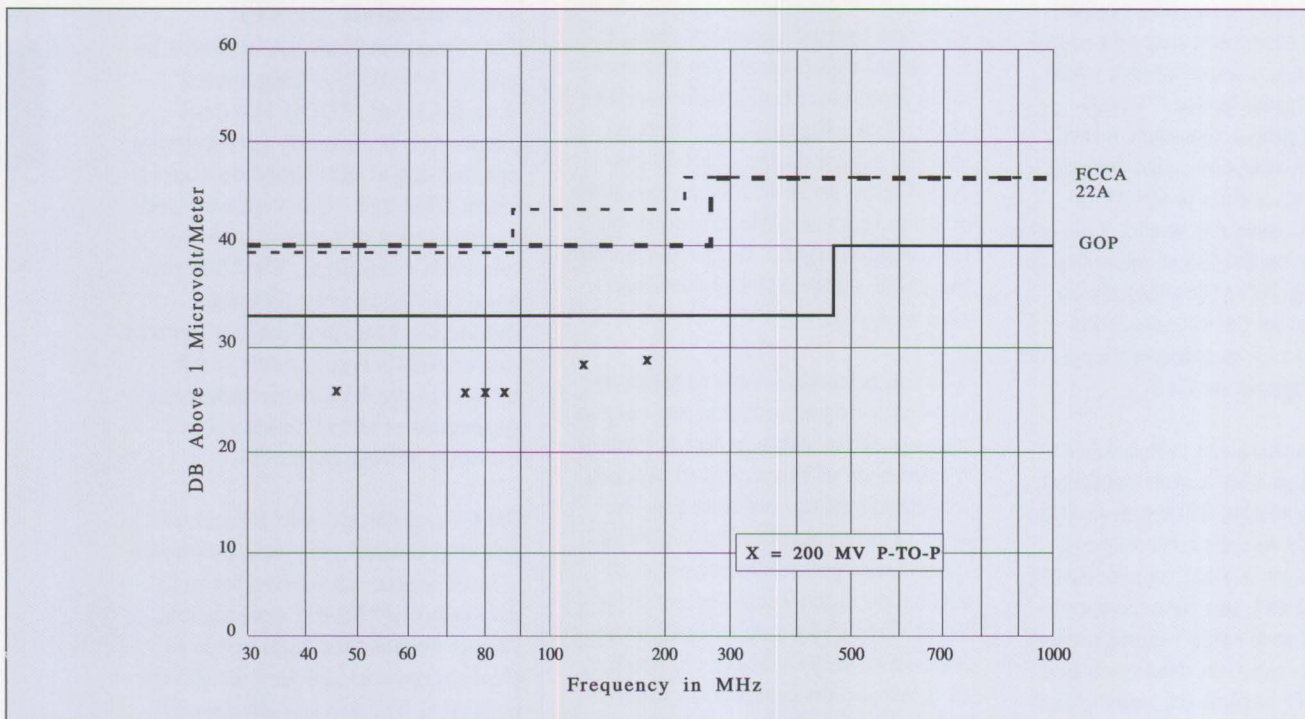


Figure 2. Radiated EMI and FCC Limits

robust and cost-effective. IBM's proposal was developed to meet the following design criteria:

- Media from the work area to the wiring closet will be 150-ohm shielded twisted-pair installed according to the recommendations in the Commercial Building Wiring Standard
- Maximum lobe lengths of at least 100 meters must be supported in accordance with the recommendations of the IBM Cabling System for work area-to-wiring-closet cabling
- Design must meet all radiation standard requirements
- Standard FDDI four-out-of-five signal coding must be retained
- Design must be robust and economical to build
- Footprint of the copper transceiver should be compatible with the optical fiber transceiver described in the FDDI standard

IBM engineers have developed a proposal that meets these design criteria. Particular attention was paid to EMI mechanisms based upon drive currents and common mode noise. The proposed solution combines a reduction in transmitter signal amplitude and the use of a wideband common mode output filter. An equalized receiver signal amplifier makes use of an optimized threshold. In addition, special consideration has been given to surge protection and signal grounding to make an environmentally robust design. All of these considerations yield a design that meets the jitter limits in the FDDI standard as well as the governmental EMI limits (Figure 1).

EMI must be considered for the whole system. However, because the proposal is for a shielded cable solu-

tion, the electromagnetic radiation is reduced by the cable's shield, which shunts the radiated signals to ground. The circuit proposed to the standard committee contains EMI common-mode filters in both the transmit and receive paths to limit further the common mode signal. The differential mode radiation is comparatively insignificant. In the relevant frequency range, that is from 30 to 200 MHz, the attenuation afforded by the filter approaches 20 dB.

Another component of EMI reduction in this design is the limit placed on the output pulse envelope. IBM is proposing that the output pulse as measured at the transceiver's D-shell connector should measure 200 millivolts peak-to-peak with a rise time of one to three nanoseconds. The receiver threshold level is set at 20 millivolts peak-to-peak. The input

pulse envelope at the receiver is expected to be 40 millivolts peak-to-peak, minimum.

The combination of the design considerations just described have been tested in the IBM Radiation Laboratory at Research Triangle Park using engineering prototypes. The graph in Figure 2 shows the EMI limits for FCC Class A, CISPR 22A, and GOP (German) certification. Our tests of engineering prototypes in a variety of configurations have met the requirements of all three of these national standards.

Just as EMI considerations affect the design of the circuit, the amount of jitter in the signal is an important component of signal quality. The FDDI Standard has a Received Optical Jitter Specification that we used as a benchmark. The graph in Figure 3 also has a line for the phase-

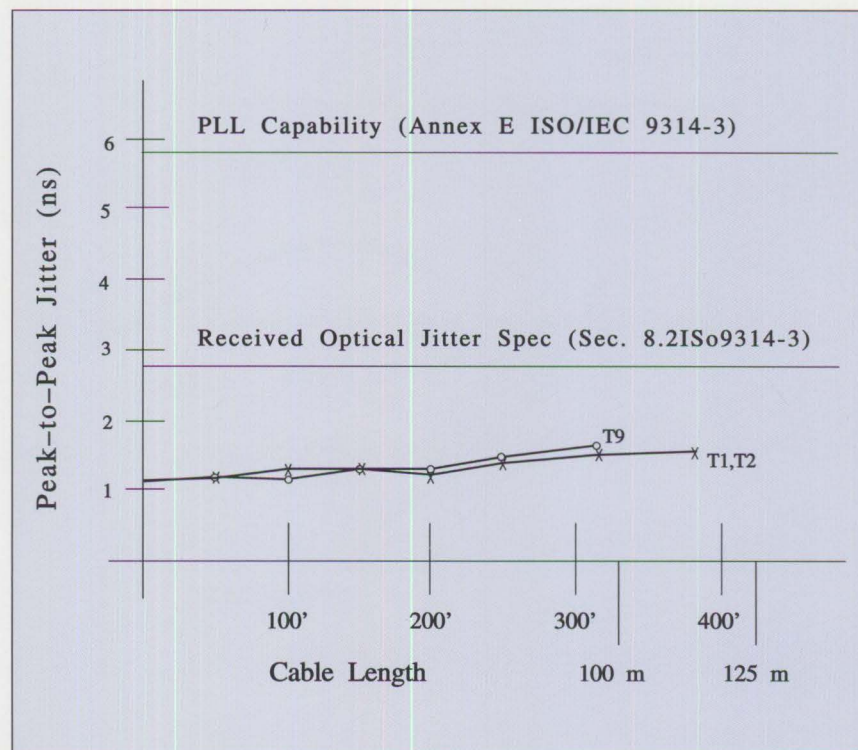


Figure 3. Jitter Test Data

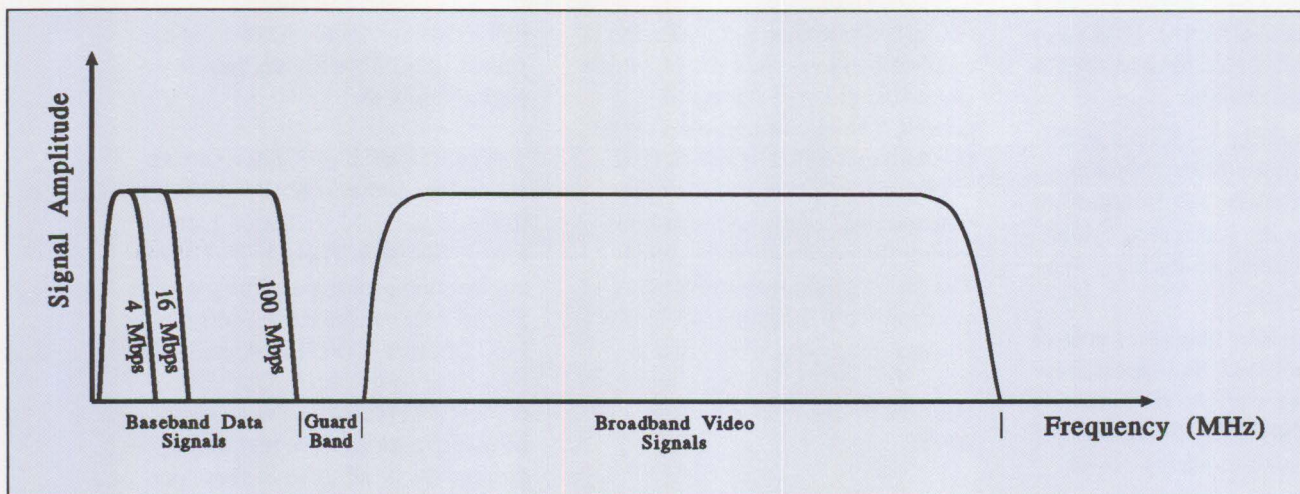


Figure 4. Signal Spectrum

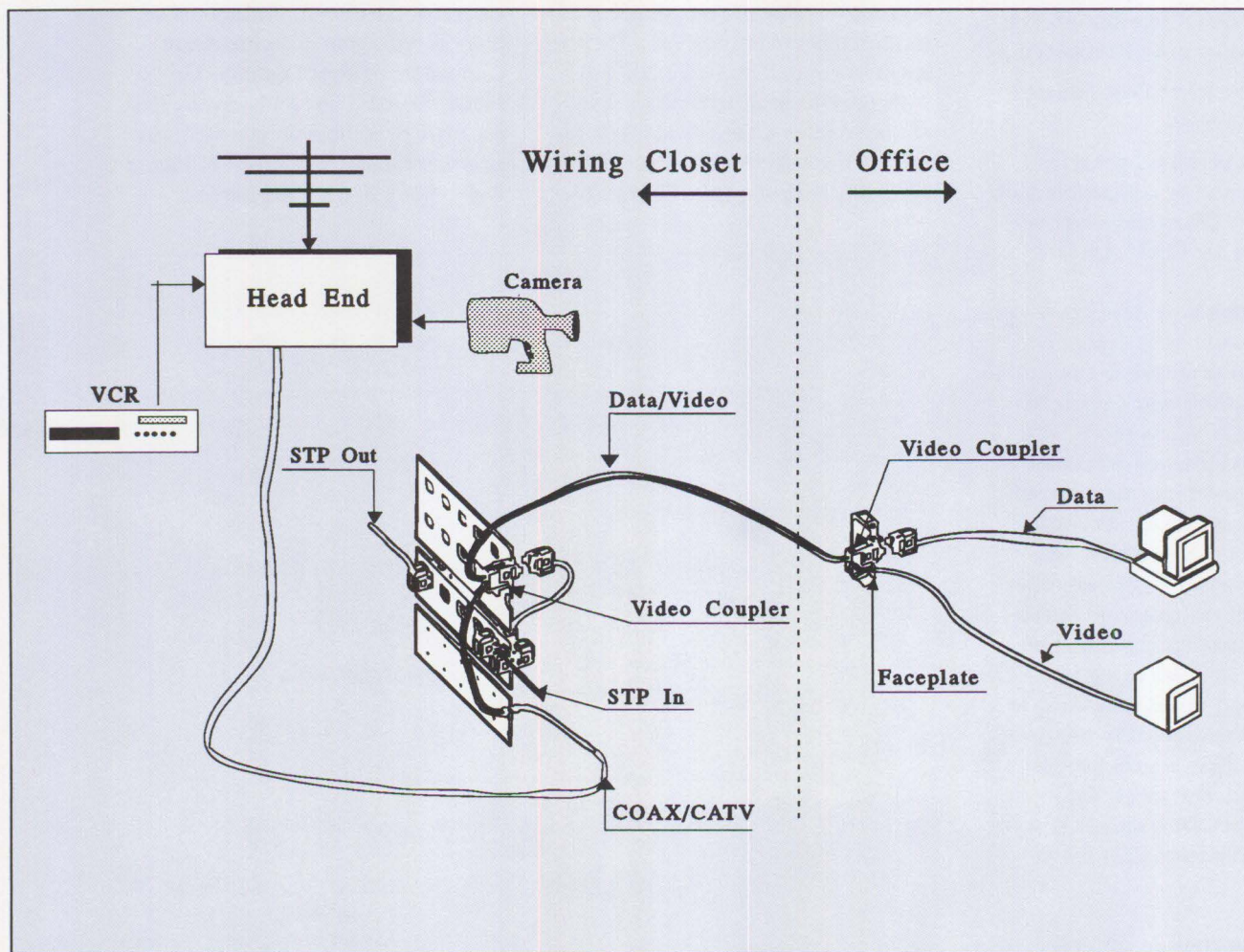


Figure 5. One Lobe Supporting Two Signals

locked loop capability described in Annex E of the ISO/IEC 9314-3 standard for comparison. As shown from preliminary test results, engineering prototypes over various cable lengths indicate that the IBM design effectively limits the jitter component of the signal. IBM engineers and other standards committee members' efforts will soon enable local area networks to operate 100 Mbps FDDI LANs on standard STP lobe cabling.

Broadband Video Capabilities of Shielded Twisted Pair

Communication requirements in most organizations grow faster than cable plants can be modified to support the new applications. Many organizations already must support multiple LANs, interconnecting LANs, and broadband video for education, training, security, imaging, and videoconferencing. Installing and maintaining a separate coaxial cable plant to serve these broadband video needs is expensive, disruptive, and may not be necessary.

Tests on shielded twisted-pair cable containing both aluminum foil and mesh shield have shown such cables to be generally uniform in their transmission parameters. The excellent shielding properties of these cables may allow simultaneous transmission of broadband and baseband signals within the same cable sheath. A video coupler, which is an experimental filtering device, has been designed to separate the two transmissions. Figure 4 illustrates the separation of these signals on the same cable. The figure shows frequency allocation occupied by the baseband data signals for the 4, 16, and 100 Mbps range. In addition, the broadband signals generally used

for video transmission ranges from 50 to 550 MHz, and higher.

For each cable run where simultaneous transmission is required, a video coupler is installed at each end of the cable, as shown in Figure 5. Each coupler would have both a coaxial connector (for broadband) and a data connector (for baseband). The circuit in the coupler isolates the broadband and baseband signals from each other and directs each of the signals to its respective port while maintaining complete isolation from each other.

By using the technique of frequency division multiplexing (FDM), greater cable bandwidth utilization can be achieved. Thus, the data and video signals each occupy only their respective bandwidths. In this way, the broadband video signal coexists with the data signal from the point where it is injected onto the cable in the wiring closet to where it is extracted at the office. This additional flexibility of STP cable should be considered when selecting cables to support expanding information requirements.

What's Next?

These new developments in our research are encouraging because they indicate new possibilities for extending the useful life of the IBM Cabling System. Just as data rates have increased faster than almost anyone would have predicted even 20 years ago, innovations in copper cable applications have accelerated to keep pace with the new demand. However, no solution is ever permanent. Demand for multimedia instructional presentations delivered to the individual's workplace are increasing; the techniques for creating these presentations are increasing in sophistication and availability.

As imaging applications begin to play a rapidly increasing role in our computing environments, we may ultimately reach the absolute bounds of either the physical or regulatory laws that govern data transmission. Optical fiber is waiting in the wings to assume its place on stage. But for a while, the stage may be shared with IBM Cabling System copper cables carrying broadband video and a 4, 16, or even 100 Mbps LAN simultaneously.

ABOUT THE AUTHORS

Hank Foglia is an advisory engineer in IBM's Communications Systems organization, presently working in the area of transmission technology development. Hank joined IBM in 1956. His experience includes introducing and developing high-speed data and broadband video transmission technology in LAN systems. He holds several patents and has authored several publications in these areas. Hank received a B.S.E.E. from City College of New York and an M.S.E.E. from Syracuse University, and is a licensed professional engineer.

Thomas Toher is an advisory information developer in IBM's Communications Systems organization. He joined IBM in 1984 and presently works in the local area network information development department. Tom represents IBM on the EIA/TIA TR 41 building cabling, infrastructure, and administration standards group. He received a B.A. and M.A. in English from Hobart College and Clark University, respectively. He is a frequent speaker on cabling and planning for token ring networks at GUIDE and other forums.

Tuning and Self-Tuning Features of OS/2 LAN Server

Tom Gordy
IBM Corporation
Dallas, Texas

While several server products on the market are advertised as self-tuning, most users don't realize that the OS/2 LAN Server has many self-tuning features. This article discusses the automatic tuning features and how users can control them through the IBMLAN.INI parameter file.

Features considered to be "self-tuning" may simply imply that the server product performs some automatic setup at initialization, and that the user has no control over what the server does. The LAN Server self-tuning features allow users to control how they are used. Such control allows tailoring of the server for your applications, if desired.

How Self-Tuning Helps Users

The OS/2 LAN Server's self-tuning features help users in many ways. For example, a user may require memory to be available to run local applications such as administrative functions, net run jobs, or command functions. If these types of memory-intensive jobs are not running, why shouldn't the server grow or expand and use the available memory? If the server doesn't need the memory, why should the memory be allocated to it? Allocating memory that will not be used is wasteful, and so is reserving memory that will be used infrequently.

BIGBUFS and Memory Utilization

The IBMLAN.INI BIGBUFS provide a way to control memory utilization. These buffers are used in several ways, but most often for downloading code or performing COPY functions. For optimum performance, there should be at least a couple of BIGBUFS available when starting a function that can use "raw data transfers." These transfers are used in COPY/XCOPY functions, code download, or as required by an application program. The question is, how many have to be set up for each user? One? Two? The answer almost always seems to be, "It depends." So, what are you going to do? Guess? That's what many people do, or they simply allocate a lot of buffers and hope. Well, the good news is, it's hard to go wrong!

SRVHEURISTICS Controls

Now don't flinch, but SRVHEURISTICS is where you go to control BIGBUFS. SRVHEURISTICS digits 17 and 18 controls how memory is allocated to BIGBUFS and how long it stays allocated. The NUMBIGBUFS parameter defines how much memory you want the server to allocate for permanent BIGBUFS. Upon startup, you get three BIGBUFS assuming you asked for three or more. Additional permanent BIGBUFS are created as they are needed. The NUMBIGBUFS parameter sets a maximum number of BIGBUFS that will be created and used for reading from and writing to the server. If users require all these BIGBUFS to be created and *still* need more to use while writing to the server, the server will create more.

Creation of "extra" BIGBUFS, known as dynamic BIGBUFS, is controlled by the SRVHEURISTICS

digits 17 and 18. Not only are they are created as needed, but released when they are no longer needed. Because allocating and creating the buffers takes cycles, you can use digit 17 to cause the server to wait for a varying period of time before releasing a dynamic BIGBUF. For best performance, this would be sufficient time to avoid the overhead whenever possible.

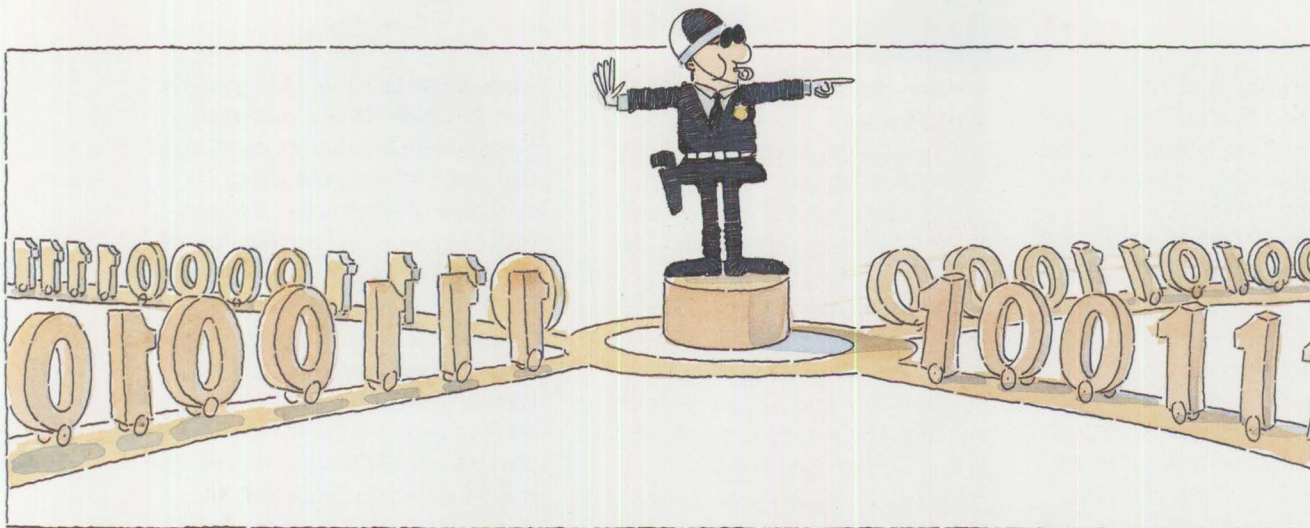
The default, **1**, releases the BIGBUF one second after it is no longer needed. You might choose to set this default to a higher value, like **3**, which waits one minute. This value keeps the BIGBUF around longer, making it available if another user or function requires it before spending the cycles to deallocate.

When a buffer is needed and no memory is available, SRVHEURISTIC digit 18 determines how often the server will try to allocate a buffer. The longer it waits, the less overhead you waste on a failure. New buffers are allocated from available server memory. If an application is using the memory, the server will wait and retry until it gets the memory required for a new BIGBUF. Trying often can get you a buffer sooner, but if the memory is not available, trying too often can slow down the system. The default, **3**, waits one minute between tries. If, during one of these waits an already created BIGBUF becomes available, it will be used and a new one will not be created.

So the server "tunes itself" for the number of BIGBUFS it has. But that's not all!

Tuning REQBUFS

SRVHEURISTIC digit 13 can be used to allocate some of the available BIGBUFS that will be used to



create more REQBUFS if the server runs out of REQBUFS, and some more buffer space is needed for prefetching data from the disk before a user requests it. These additional buffers are called "4 KB read-ahead buffers."

Have you ever seen the NET ERROR message that indicates that the server ran out of the resource defined by the NUMREQBUFS parameter? There's also a message for NUMBIGBUFS. If you enter the NET STAT SRV command, the last two lines of the server statistics screen show exactly how many times this has happened. By setting digit 13 to 9 (and having at least that many BIGBUFS), you can give the server an additional 144 buffers to be used for prefetching data.

Because the server creates these REQBUFS as required, this is another example of (controlled) automatic tuning of the server. If you allocate nine BIGBUFS to this function, you won't need 144 more NetBIOS commands before they can be used. You couldn't define that many extra commands anyway. These buffers are used to hold data that requesters haven't yet asked for, but which the server expects them to

ask for soon. The data is not actually sent to the requesters until they ask for it. Having the data waiting in a buffer when a requester asks for it means the requester will get the required data much faster than if the server had to read it from the disk. SRVHEURISTIC digit 13 defaults to 1. That's enough memory to create 16 additional REQBUFS to hold data as required. If you have more than 20 users, you might increase this value. In fact, you might want to increase it by one for every 40 users after that.

Tuning Cache

Another feature designed to help server performance is the way the server uses its cache. The cache is designed to allow many users to share frequently accessed data. In other words, the data is used in applications that share files and do byte range "record" locking. That, in turn, indicates that the data is randomly accessed (probably by a data base).

For these reasons, the OS/2 caches (DISKCACHE for FAT drives, or HPFS cache.exe for its drives) are designed to *not* cache sequential file accesses in OS/2 Version 1.3. If sequential file I/O was cached, your cache could quickly fill with useless

data – print files going to or from the spool queues, for instance, or programs that have been downloaded.

It would be nice to support more cache, but the HPFS cache is limited to 2 MB, and the DISKCACHE can be no larger than 7.2 MB. There are many things to consider before changing the way cache is used. For example, if it takes one second to cache 200 KB of data, how many times would the data have to be re-used before the cache actually saved time overall? Is it likely the data will be used that many times before it is overlaid by newly cached data? Will the new data be used enough before it, in turn, is overlaid by some even newer data? These are not easy questions to answer.

It may be nice to keep some of this sequential stuff in memory. For example, RIPL time can be halved if the RIPL image is memory-resident in the server. Program loads would be faster if programs were in memory when needed. To accomplish this, create the VDISK when the system is started, copy the code to it in STARTUP.CMD, and share the VDISK at server startup. Then users can get programs from memory.

If you're using the FAT file system and DISKCACHE, you have the option of defining the maximum disk read length that will be cached. By default, this is 3.5 KB. Using the parameters on the CONFIG.SYS DISKCACHE statement, you can increase this so disk reads up to 32 sectors will be cached (DISKCACHE = xxxx,32). This could be handy for sequentially accessed files if they could be cached when the first record is read. HPFS cache does not have this capability, and it will cache no reads over 2 KB (eight sectors). Therefore, HPFS is the best place to put your randomly accessed data files. If both file systems and caches are used, you could have up to nearly 10 MB of cache. Be sure to check that your cache (or VDISK) does not force the server into a swapping mode. If that happens, all tuning efforts will be wasted.

Some users have found "another cache" in the INI file — MAXWRKCACHE, which is Microsoft's term for the requester's BIGBUFS. While it might seem strange to specify a number of buffers on the server and a total amount of memory (in 64 KB increments) at the requester, that's the way it's done. At any rate, MAXWRKCACHE isn't a cache except in a very limited sense. If you're using sequential I/O and start using BIGBUFS of data (the raw data transfers mentioned earlier), then the WRKCACHE is a cache for a *single* file. Actually, it's more like reading a big block of data from a tape drive or something similar.

Effect on Slow Bridges

Okay, enough on caches. Let's look at another tuning option to consider. LAN servers sending large (64 KB) messages on slow bridges (either

bridges on slow telecommunications links or local bridges that are suffering congestion) have caused problems when the NetBIOS at the server timed out and disconnected the remote user. This occurs if the bridge could not forward data as fast as the server could send it. When this occurs, users see a SYS0240 error message. Various tuning changes have been used to help this situation (for example, changing the 802.2 T1 timer and setting NUMBIGBUFS to zero), but nothing cured it completely. And, there is no NetBIOS timer parameter to tune.

There *is* a way to change the NetBIOS timer. SRVHEURISTICS digit 15, which determines the OPLOCK timeout, is also used for the NetBIOS timeout. Unfortunately, it's not all that simple, because OPLOCK timeout is a 16-bit number and NetBIOS timeout is an 8-bit number. Therefore, the NetBIOS timeout can be no higher than 255 (and that is in .5-second intervals, so 255 is really 127.5 seconds). Stuffing a 16-bit number into an 8-bit field changes its value in some cases because the result is modulo 255. To save you some arithmetic, here are the SRVHEURISTIC digit 15 values, the OPLOCK timeouts, and the NetBIOS timeouts:

Digit 15	OPLOCK	NetBIOS
0	35	17.5
1	70	35
2	140	70
3	210	105
4	280	12
5	350	47
6	420	82
7	490	117
8	560	24
9	640	64

These times are all in seconds; conversion is not needed.

To net this out, if you have the LAN Server 1.2 or 1.3 and slow (or congested) bridges, and if you are having problems with the server dropping NetBIOS sessions with requesters on the "other" ring, set the SRVHEURISTICS digit 15 to 1, not the 0 default. If problems persist, 7 is the greatest value you might try. If you do this, be aware that the OPLOCK time for a value of 7 is 490 seconds. That is the time an application would have to wait before receiving an "access denied" message if the OPLOCK holder terminated abnormally.

The good part is that if either server (1.2 or 1.3) drops the requester, the next time the requester tries to access the server, the dropped session is re-established automatically. Therefore, even if the user cannot perform the "slow" function, other jobs can continue on the requester or the function can be retried later.

Summary

The IBM LAN Server has several self-tuning features. Understanding how to control them through the IBMLAN.INI parameters can help you make the best use of the server's resources in your installation.

ABOUT THE AUTHOR

Tom Gordy is a senior market support representative in IBM's LAN Cross Systems Solutions Support in Dallas, Texas. He is responsible for LAN Server performance, capacity, and tuning support. Tom has been a presenter at IBM television broadcasts to its customers and has published several IBM technical bulletins.

NetWare Communications and Routing Protocols

Paul Turner
Novell Inc.
Provo, Utah

In the early days of local area networks (LANs), the networks were simpler and usually had only one file server. People didn't know about the communication processes between workstations and file servers. The LANs of today are much more complex with multiple servers and different types of protocols being used. Understanding how data moves through the network has now become essential. This article explains how NetWare gets information from one place to another on a network.

Routers perform a vital role in networking. Their function is far more complex than simply passing packets from one network segment to another. Routers share information with other routers on the network and are the focal point for disseminating information on their particular segment. To understand the various roles routers play requires an understanding of the mechanics of NetWare communications.

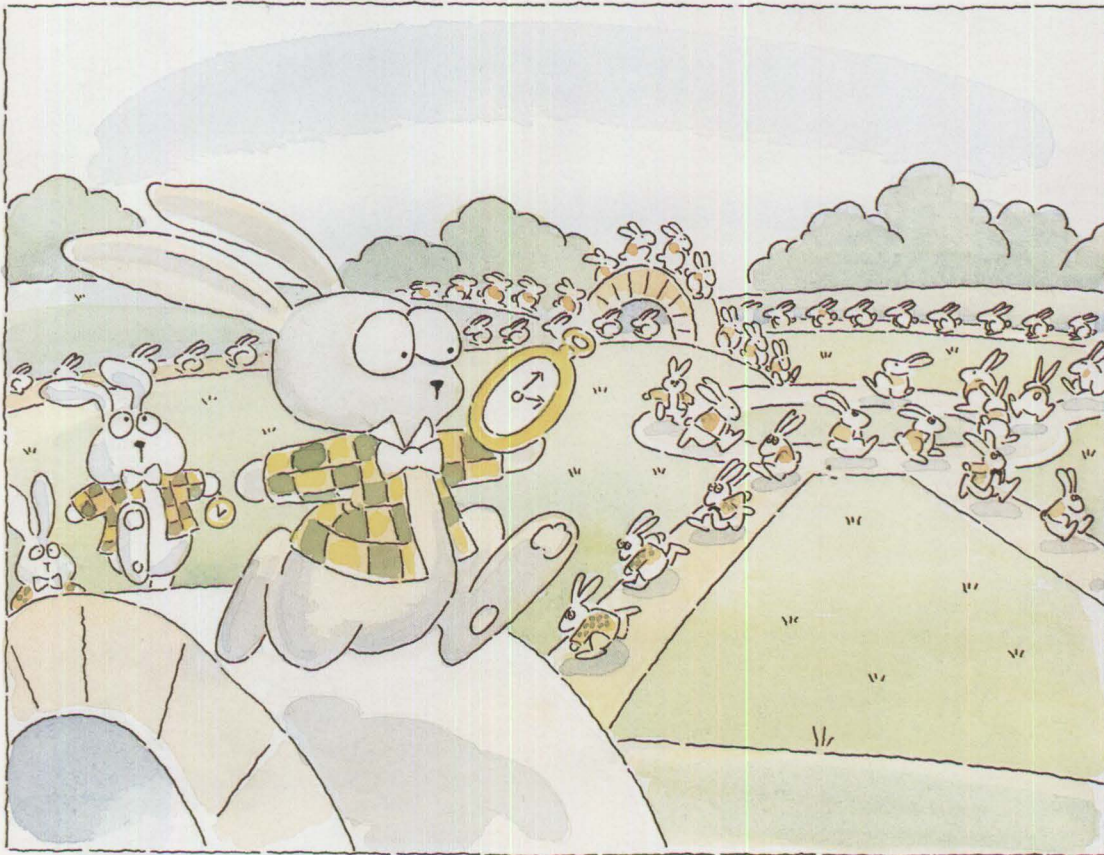
When networks were simple systems with one file server on a single trunk cable, designers and installers didn't need to know much about the communication processes that interacted between workstations and file servers. They spent more time trying to get single-user software packages to run in a multi-user environment than worrying about internetworking issues.

Since then, networks have become larger and more complex. Products such as multiprotocol routers and MAC-layer bridges, and protocol implementations such as token ring source routing and TCP/IP, are becoming necessary elements in the design of large internetworks. To successfully integrate these and other elements with NetWare, a solid understanding of the mechanics of NetWare's communications environment is essential.

Note that what Novell has traditionally called *internal bridges* (those within file servers) and *external bridges* are referred to as *internal* and *external routers* to be consistent with standard industry terminology.

Protocols for Exchanging Information

Any in-depth discussion of the mechanics of NetWare communications



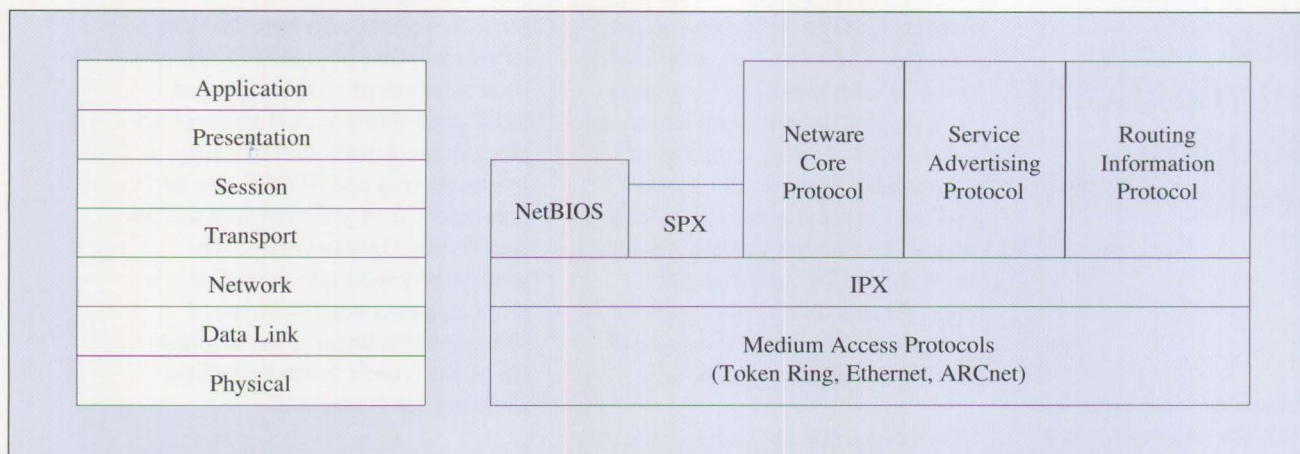


Figure 1. Relationship of NetWare Protocols to OSI Model

requires a good understanding of the various protocols NetWare uses. These protocols define both the language and the packet format used to exchange information over the network.

Basically, seven different protocols are used for communication and packet routing within the native NetWare environment. These include:

- Medium access protocols
- Internet packet exchange (IPX)
- Routing information protocol (RIP)
- Service advertising protocol (SAP)
- NetWare core protocol (NCP)
- Sequenced packet exchange (SPX)
- NetBIOS

Figure 1 shows how these protocols relate to the layers in the open systems interconnection (OSI) reference model. Although there is not a direct correlation in layer boundaries between the two architectures, this relative mapping gives you a general idea of where each protocol fits.

As in the OSI model, the upper NetWare protocols (RIP and SAP)

rely on the lower protocols (IPX and the medium access protocols) to take care of the lower-level communication issues. This modular approach is the key to NetWare's compatibility with numerous network interface cards.

Our discussion focuses on the first four protocols, because they deal most closely with NetWare communication and routing. NCP, SPX, and NetBIOS are more concerned with peer-to-peer and client/server interaction.

Medium Access Protocols

Medium access protocols are defined by the network hardware being used. A number of these protocols have been developed, many of which can be used with NetWare. We'll concentrate on the most common medium access protocol implementations: 802.5 token ring, 802.3 Ethernet®, Ethernet version 2.0, and Arcnet.

These implementations are primarily concerned with transporting packets from one node to another on a single network segment. They provide bit-level error checking in the form of a cyclic redundancy check (CRC).

The CRC is appended to every packet transmitted, thus assuring that 99.9999 percent of the packets that are successfully received are free of corruption. In view of this level of integrity, NetWare does not provide any additional bit-level error checking within any of its upper-level protocols.

NetWare's Addressing Scheme:

The basis of any communication and routing methodology is the addressing scheme employed to distinguish different entities on an internetwork. NetWare's addressing is analogous to the postal service's addressing system that identifies mail recipients by country, state, city, street, number, apartment, and individual names. The postal system can adopt the addressing conventions of an apartment complex into its overall system. Similarly, one protocol can integrate another's addressing scheme into its own – as in the case with IPX and the medium access protocols.

The medium access protocol implementations define the portion of the NetWare address that distinguishes each node on a network. Node addressing is implemented within the hardware of each network interface

File Servers	451h – Reserved for NetWare Core Protocol
Routers	452h – Service Advertising Protocol 453h – Routing Information Protocol
Workstations	4003h-4007h – Communication with File Servers 455h – NetBIOS

Figure 2. Socket Number Uses

card (NIC). To get a packet to the proper node on a network, a medium access control (MAC) header is placed at the front of every packet. The MAC header contains a source address field and a destination address field that indicate where the packet originated and where it is going. If the destination address in the MAC header matches a NIC's own node address, or if the packet is

identified as a broadcast packet (intended for all nodes), the NIC makes a copy of the packet.

The majority of medium access protocol implementations provide bit-level error checking and node addressing. IBM's token ring implementation defines a method of routing called *source routing*, which allows network traffic to be seg-

mented by separating rings with IBM token ring bridges. This requires that each workstation maintains a table containing routes to each node they are communicating with. To maintain these tables, routing information must be included in the MAC header of each packet sent. Source routing can be used instead of, or in conjunction with, NetWare routing, as long as the network is laid out properly.

Internet Packet Exchange

Novell adopted its IPX protocol from the Xerox® Network System (XNS). IPX is a datagram-based, connectionless protocol. It is connectionless because it does not require an acknowledgment for each packet sent. This packet acknowledgment, or connection control, must be provided by protocols above IPX.

IPX defines internetwork and intra-node addressing schemes relying on the medium access protocol for node addressing. The network numbers assigned in NETGEN for NetWare v2.1x form the basis of IPX's internetwork addressing. Each network segment on a NetWare internetwork is assigned a unique network number. NetWare routers use this number to forward packets to the final destination segment, much like the postal service uses a street name. Once a packet reaches the final destination segment, the node address is used to deliver the packet to the proper workstation or server.

The IPX address for routing packets within a node comes in the form of socket numbers, which provide a quick method of routing packets within a node. Because several processes are normally operating within a node, sockets provide sort of a mail slot so each process can distinguish itself to IPX. When a process

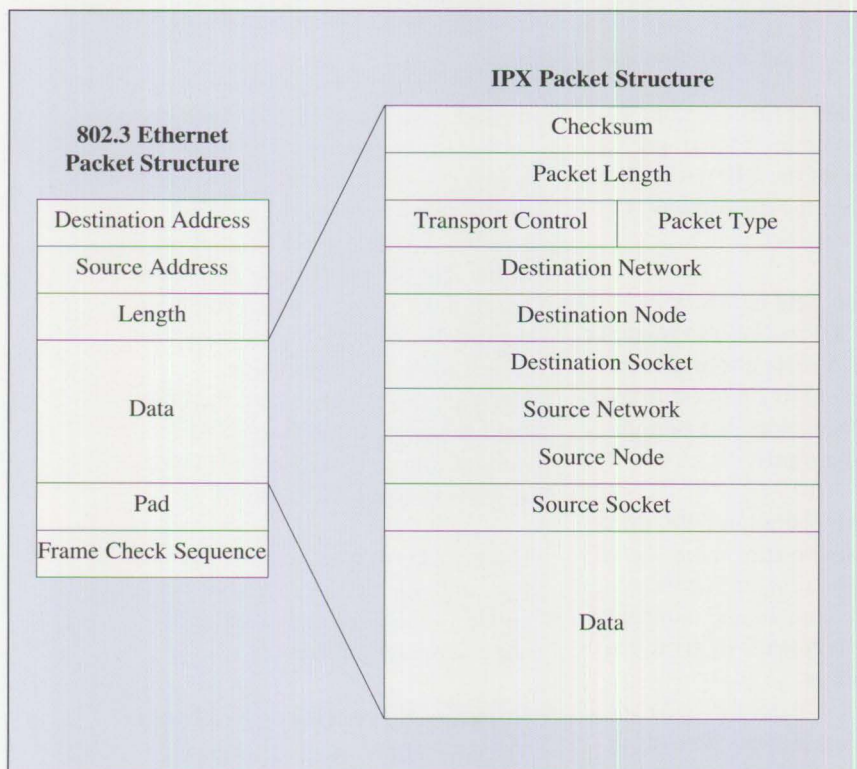


Figure 3. Structure of an IPX Packet (802.3 Ethernet)

needs to communicate on the network, it requests that a socket be assigned to it. Any packets that IPX receives that are addressed to that socket are passed on to the process. Novell has reserved several socket number for specific purposes (Figure 2).

The network, node, and socket addresses for both the destination and source are held within the packet's IPX header. This header is placed after the MAC header and before the data packet. (A packet's data can include the headers of higher-level protocols.) Figure 3 shows the structure of an IPX packet on an 802.3 Ethernet network.

Routing Information Protocol

The RIP facilitates the exchange of routing information on a NetWare internetwork. Like IPX, the RIP was derived from XNS. However, a critical change was made in the packet structure to improve the decision criteria for selecting the fastest route to a destination. This change prohibits the straight integration of NetWare's RIP with other, undeviating XNS implementations.

The single-packet structure defined by the RIP allows the following information exchanges to take place:

- Workstations can locate the fastest route to a desired network number
- Routers can request information from other routers in order to update their internal tables
- Routers can respond to requests from routers and workstations
- Routers perform periodic broadcasts to ensure that all other routers are aware of the current internetwork configuration

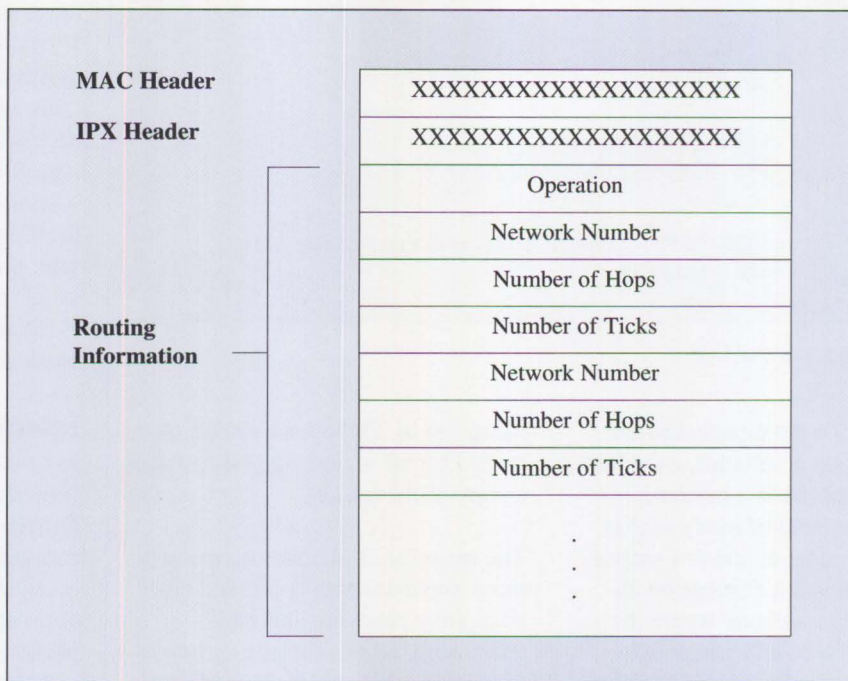


Figure 4. Routing Information Protocol Packet Structure

Each of these information exchanges will be discussed in more detail with the mechanics of NetWare routing.

RIP Packet Structure: Figure 4 shows the structure of a RIP packet. The structure of the RIP packet is enveloped within the data area of a standard IPX packet.

The operation field indicates whether the RIP packet is a request or response. A response packet can be either a reply to a request from a router or workstation, or a periodic broadcast by a router.

Following the operation field, the packet can hold one or more sets of routing information, each consisting of a network number and number of *hops* and *ticks* it takes to get to that network number.

A hop is counted every time a packet passes through a router to reach a network number. Routers in-

clude themselves as one hop when counting the total number of hops.

A tick is roughly one-eighteenth of a second. (There are 18.21 ticks in a second, to be precise.) The total number of ticks measures how much time it takes to reach the network number. The original XNS definition of the RIP did not include the "number of ticks" field; it relied solely on the number of hops to determine the fastest route to a destination. NetWare developers added the time field to integrate NetWare with a variety of transport mechanisms (such as asynchronous, X.25, and T1).

If the packet is a request for information, only the network number field applies – the hops and ticks fields are essentially nulled out.

Service Advertising Protocol

The SAP allows nodes that provide a service (like file servers, print servers, and gateway servers) to adver-

tise their services and addresses. The SAP is an adaptation that makes the process of adding and removing services on a network more dynamic.

Through the SAP, clients on the network can determine what services are available and obtain the inter-network address of the nodes (servers) where they can access those services. This is an important function, because a workstation could not initiate a session with a file server without first obtaining the server's address.

For example, a gateway server broadcasts a SAP packet every 60 seconds – a period defined for all servers advertising with the SAP – onto the network segment to which it is connected. Each router on that network segment copies the information in this SAP packet into an internal table called the server information table. Because each router keeps up-to-date information on available servers, a client wanting to locate the gateway server can access a nearby router for the address.

Like the RIP, the SAP uses IPX and the medium access protocol for its transport. Figure 5 illustrates the SAP packet structure.

Again, the first field after the IPX header defines the operation the packet is performing. There are four possible operations:

1) A request by a workstation for the name and address of the nearest certain type of server. For example, the workstation might essentially say, "Give me the name and inter-network address of the nearest file server." This request would be represented by a "get nearest server" entry on the NetWare TRACK ON screen. (TRACK ON is a utility available at the file server or exter-

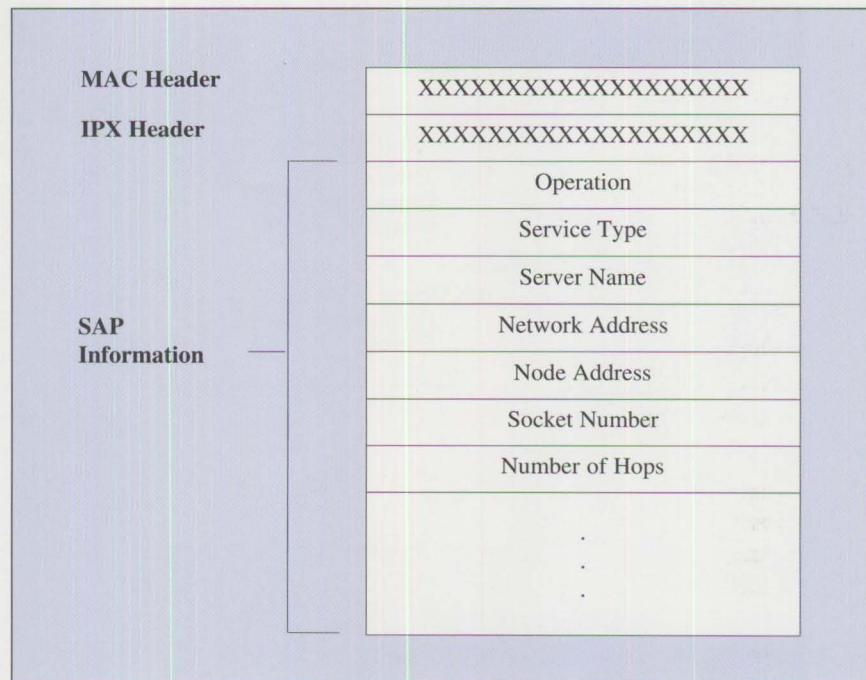


Figure 5. Service Advertising Protocol Packet Structure

nal router console that allows an administrator to track RIP and SAP exchanges on the network. It is accessed by typing "track on" at the colon (:) prompt.)

(2) A response to a Get Nearest Server request. (This response would be seen as a "give nearest server" entry on the TRACK ON screen.)

(3) A general request for the names and addresses of all servers, or all servers of a certain type, on the inter-network (seen as "Send All Server Info" on the TRACK ON screen).

(4) A response to a general request.

Following the operation field are one or more sets of SAP information fields, each containing the following information about a server:

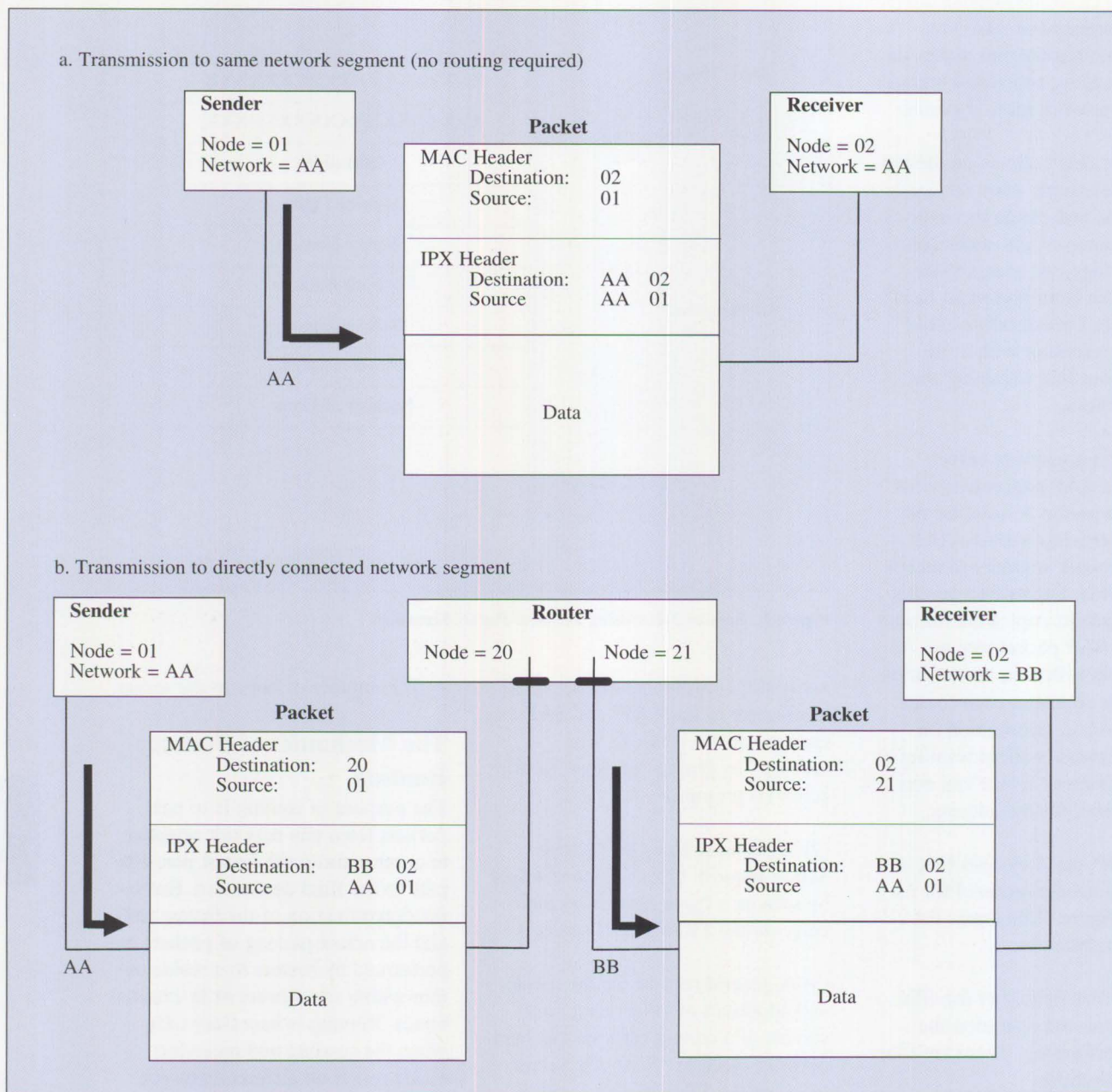
- A number representing the type of service provided by the server
- The server's name and address

- The number of hops to the server

The Mechanics of Packet Routing

The purpose of routing is to pass packets from one network segment to another using the fastest possible path to the final destination. Both the determination of the fastest path and the actual passing of packets are performed by routers that reside either within file servers or in external boxes. Routing is necessary only when the sending and receiving nodes reside on different network segments separated by one or more routers.

Sending a Packet: The sending node must have the full address (network, node, and socket) of the destination node before sending the packet. The sender places this address in the destination address fields of the IPX header. It also places its own address in the source



Figures 6a and b. How MAC and IPX Headers Are Used in Routing

address fields so the receiver knows to whom it should respond.

If the destination network number is the same as its own, the sender places the same set of destination and source node addresses in the

MAC header and transmits the packet (Figure 6a).

If the packet is destined for a different network number, the sender has to send the packet to a router that can forward the packet. It is not necessary for the sender to know the en-

tire route to the destination, as in source routing. It only has to locate the router on its local network segment that registers the shortest path to the destination network number and send the packet there.

The sender finds the appropriate router by locally broadcasting a request for the shortest route to the destination network number. (Local broadcasts are not forwarded to other network segments by routers.) The router with the shortest route responds to the request giving its own node address, plus the total number of routers (hops), and total time in eighteenths of a second (ticks) necessary to reach the desired network number.

To send the packet, the sender places the node address of the router in the MAC header's destination address field and its own node address in the source address field, leaves the IPX header as previously set, and transmits the packet. When the router receives the packet, it checks the destination address fields of the IPX header to determine what action should be taken (Figure 6b).

At the Router: NetWare routers maintain a routing information table that holds information about all of the network segments on the internetwork. Each entry in this table tells the router how to forward packets so they reach a particular network segment. With this information, the router acts like the flight attendant who directs you to your connecting flight as you deplane.

The network segments are arranged by network number. The router simply matches the destination network number in the packet's IPX header with an entry in its routing information table to get its forwarding instructions. There are three possible instructions for forwarding the packet:

(1) If the router is an internal router and the packet is addressed to that file server, the packet is passed di-

rectly to the file server for processing.

(2) If the packet is destined for a network number to which the router is directly connected, the router places the destination node address from the IPX header in the destination address field of the MAC header, places its own node address in the source address field of the MAC header, and transmits the packet.

(3) If the router must send the packet to another to reach the destination network number, the router obtains the next one's node address from its routing information table. It places that address in the destination address field of the MAC header, places its own in the source address field, and transmits the packet on the appropriate NIC.

It's important to realize that the IPX destination and source address fields have not been changed during any of these examples. They remain the same for two reasons: so subsequent routers (if any) can follow the same algorithms as the first router did, and so the final destination node knows to whom to respond. Also, the packet does not keep a record of the path taken to reach its destination. The only modification that the router makes to the IPX header is incrementing the transport control field, which counts how many routers the packet has passed through.

Spreading the Routing Information

Now that we've covered the process of routing, let's talk about the administrative processes going on in the background. On an active internetwork, routers are constantly exchanging information with each other to make sure that their routing information tables reflect up-to-the-minute

changes in the layout of the internetwork. To maintain current information in their tables, routers transmit a series of broadcasts (using the RIP) from the time they come up until they are eventually brought down. These broadcasts can be separated by the time at which they occur:

(1) First, there's an initial broadcast of directly connected network segments

(2) Next, there's an initial request to receive routing information from other routers

(3) After these initial broadcasts, routers perform periodic broadcasts (every 60 seconds) of the current list of active network numbers

(4) When necessary, routers send broadcasts notifying other routers of a change in the internetwork configuration

(5) When a router is brought down, it issues a final broadcast

Although broadcasts occur at different times, and for the most part contain different information, they must follow two important guidelines. First, each broadcast must be a local broadcast, addressed so it will not be immediately passed on intact by the routers that receive it. This reduces the network load created by these information exchanges. Second, routers must follow a *best information algorithm* when providing information to other routers through a broadcast. (Requests for information, such as the second broadcast previously listed, are not subject to the best information algorithm.)

The Best Information Algorithm: The purpose of routing information broadcasts is twofold: to allow a router to share its current impression

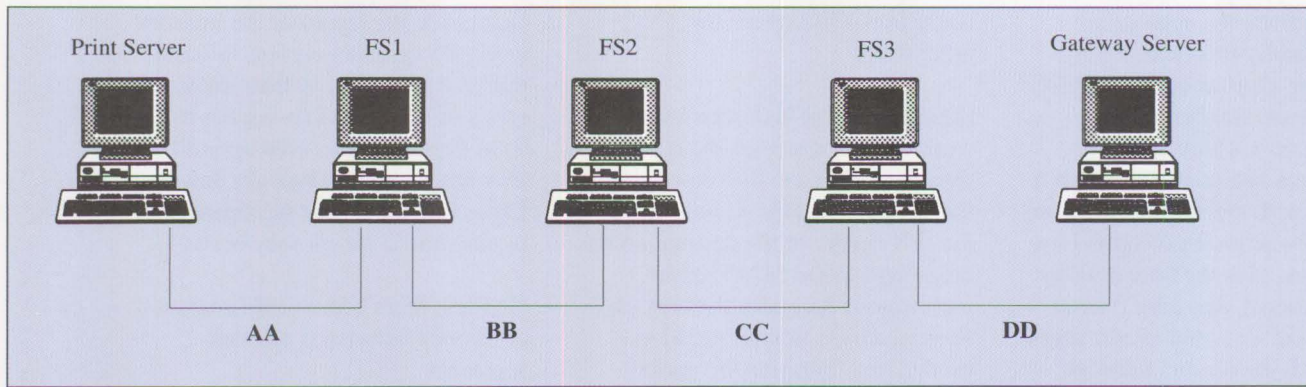


Figure 7. Sample Four-Segment Internetwork

of the layout of internetwork with other routers, and to inform other routers of a change that can update their tables.

A router sends routing information broadcasts to every network segment to which it is directly connected. The first rule of the best information algorithm dictates that a router about to broadcast onto a particular network segment should not include any information about other segments it has received from that segment. For example, if the router within server FS2 in Figure 7 is going to send a routing information broadcast onto network segment BB, it should not include information it received from FS1 about network segment AA. If it did, someone on segment BB might erroneously conclude that there are two paths to segment AA (one through FS1 and another through FS2).

This algorithm also states that routers should not include information about the network segment to which information is being sent. For example, FS2 would not include information about BB in its broadcast onto BB. Therefore, the information that FS2 broadcasts onto segment BB would be information about segments CC and DD.

Routing Information Broadcasts:

When a router is first brought up, it places network numbers of its directly connected segments into its routing information table. Then, following the best information algorithm, the router sends a routing information broadcast to inform the routers on its directly connected segments of the segments it will now make available. The router next broadcasts a request onto each of its directly connected segments for information about all other network segments on the network. All routers on its directly connected segments respond to this request (each using the best information algorithm). The router places the information gleaned from these responses in its routing information table. Figures 8a, b, and c illustrate this sequence of initial broadcasts.

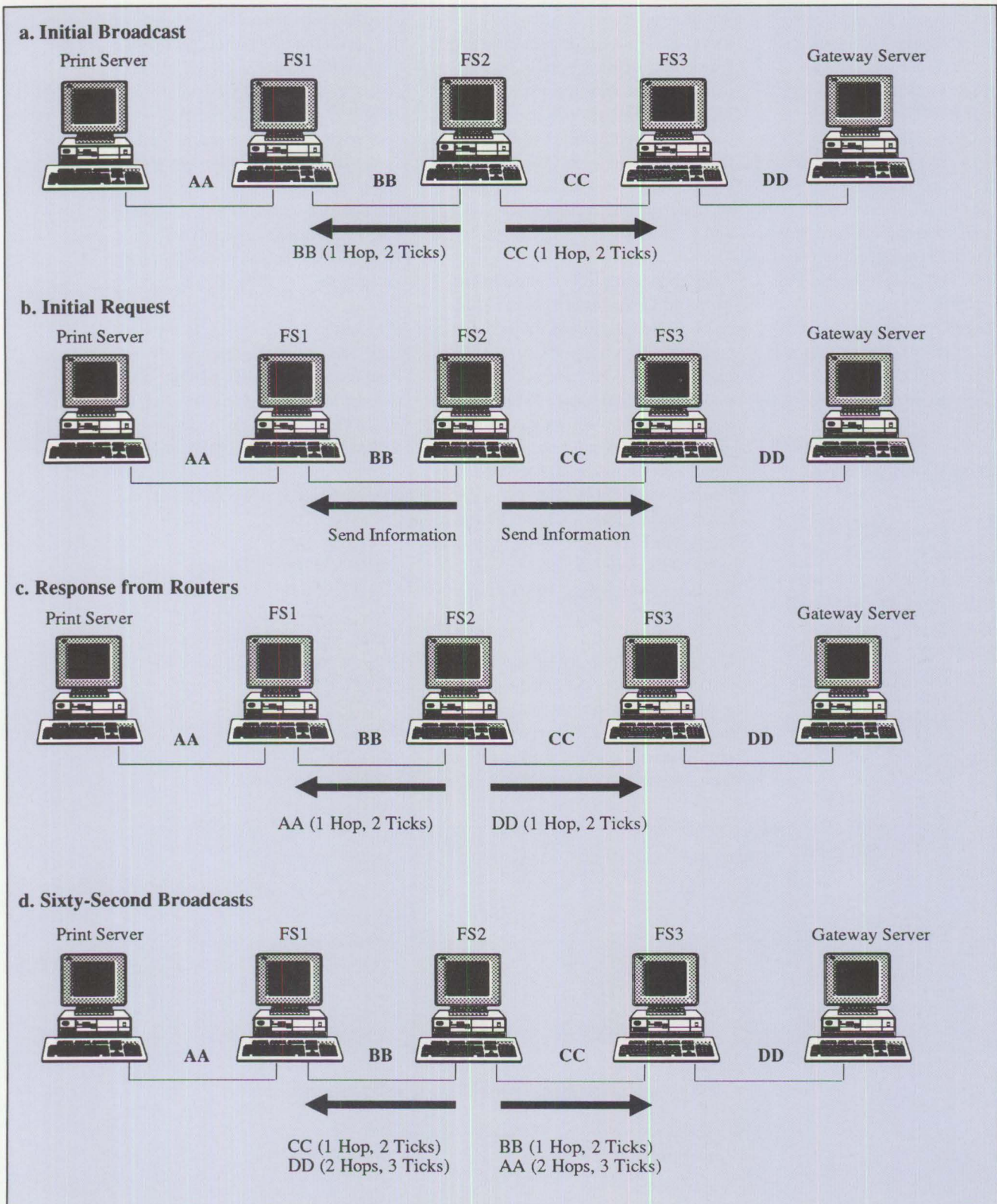
Once a router has performed these initial broadcasts and updated its routing information table, it is ready to route packets. In addition to routing packets, every 60 seconds the router broadcasts all the information in its routing information table (except that excluded by the best information algorithm) onto each of its connected network segments. Routers perform these periodic broadcasts to ensure that all routers on the internetwork remain synchronized.

Figure 8d shows an example of such a broadcast. (Because of their low bandwidth limitations, X.25 and asynchronous links do not perform 60-second broadcasts.)

Changes on the Internet: When a router receives information that causes it to change its routing information table, it immediately passes that information to its other connected network segments. That is, all segments except the one on which received the new information. Therefore, if a new network segment comes up or an existing one goes down, all routers on the internetwork hear about it quickly.

If a router needs to be brought down (via the DOWN command at the console), it informs its peers before discontinuing service. Using the best information algorithm, it issues broadcasts indicating that the network segments it made available will no longer be accessible through this router – though they may still be available through another router (Figure 9).

On the other hand, if a hardware failure, power glitch, or – though this never happens – a user turning it off, causes a router to go down *without* the DOWN command, other routers will not immediately be aware that a



Figures 8a, b, c, and d. Building and Maintaining a Routing Information Table

change has occurred. To safeguard against this, an "aging" mechanism is built into NetWare routers. Routers maintain a timer for each entry in their routing information table. Every time information is received concerning the entry, the timer is reset to zero. If the count on the timer reaches three minutes, the router assumes that the route to the network number referenced by that entry is down and broadcasts that to its other segments. Because this is new or changed information, routers that receive it will pass it on immediately and news of the change quickly permeates throughout the internetwork.

Advertising Services with SAP

The method of broadcasting previously described for distributing routing information across an internetwork is also used for circulating server information. As servers broadcast their services and addresses with SAP, routers collect the information in their server information tables. They will subsequently pass on this information to other routers through SAP broadcasts.

An internal router residing in a server performs an initial server information broadcast and a request for server information from other routers updates its server information table. Thereafter, the internal router performs broadcasts about the servers that it is aware of every 60 seconds (except on asynchronous and X.25 links).

As with routing information broadcasts, all server information broadcasts are local and are subject to the best information algorithm. Any changes in server information are passed on immediately to ensure current information across the internetwork. The router applies the aging process for its server information table entries in case any servers have become unavailable. Finally, as the server is brought down, it indicates to its peers that the servers it has been advertising are no longer available.

Using the SAP, a workstation can broadcast a request onto the segment it resides on to get the name and address of the nearest server of a certain type. All routers with access to the closest server on the segment re-

spond to this request. If several internal routers exist on the network, they will answer with the name of the server where they reside (registered as one hop away). Any external routers on the segment will not answer, because all servers within their tables will be at least two hops away. The workstation broadcasting the request acts on the first response it receives, ignoring all subsequent responses.

Once the workstation has the name and address of the nearest server, it can request the fastest route to the server and try to establish a connection. Once connected to a file server, the workstation can query that server's bindery for the address of the server that it wants to log onto.

ABOUT THE AUTHOR

Paul Turner is a consultant with the systems research department of Novell's Systems Engineering Division.

*Novell Inc.
P.O. Box 5900
Provo, Utah 84606*

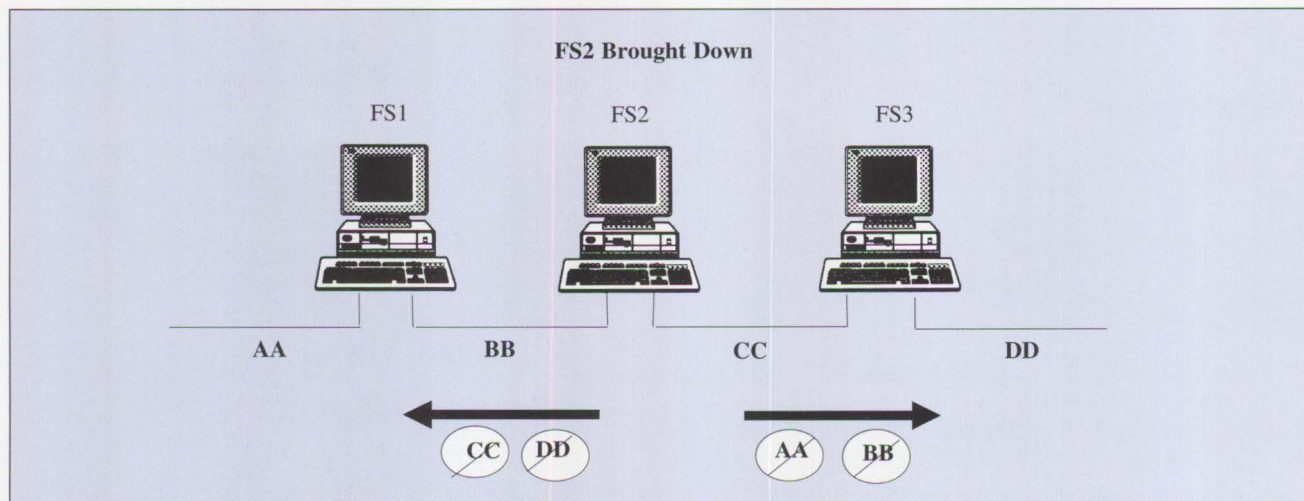


Figure 9. Routers Inform Other Routers When Going Down



Little Solutions for LANs

Our *Little Solutions* column in this issue is a collection of hints and tips that will be helpful to those installing LANs.

NETLOGON Service

The NETLOGON service runs on all servers. This function must run on the domain controller for users to logon onto the domain. The primary task of NETLOGON is to validate users logging onto the network. The other major task is to update the NET.ACC on the other servers in the domain. The NETLOGON ser-

vice must run on an additional server to get any updates to the NET.ACC file. This file contains all user IDs and their passwords, group names, and access profiles.

NET8002 After Installing a CSD

When the OS/2 Communications Manager is installed, default configuration files ACSCFG.CFG and ACSCFGUS.CFG are furnished. When a CSD is installed, new versions of these files are copied over current ones. Installation instructions recommend that a copy of these files should be used with different names. If default files are used, current files are replaced with new ones, and all information in the old files is lost.

The configuration file may get written over on a server network during CSD installation. The server will fail to start after the CSD is applied and the error message NET8002 is then displayed. If Communications Manager is used for host communications, it must be reconfigured.

Backing up NET.ACC File

Here's an easy way to keep a current backup copy of your NET.ACC file. Figure 1 contains an example of lines that could be used in the STARTUP.CMD. These three lines store copies of the NET.ACC from the last three reboots of the server. Use whatever file names you want for the backup files. Keep a NET.ACC backup for each server.

```
COPY C:\BMLAN\ACCOUNTS\NETBACK2.ACC C:\BMLAN\ACCOUNTS\NETBACK3.ACC
COPY C:\BMLAN\ACCOUNTS\NETBACK1.ACC C:\BMLAN\ACCOUNTS\NETBACK2.ACC
COPY C:\BMLAN\ACCOUNTS\NET.ACC C:\BMLAN\ACCOUNTS\NETBACK1.ACC
```

Figure 1. Backup of NET.ACC in STARTUP.CMD File

Users Unable to Get Access to Shared Resources

If you can't access a resource that should be available, there's an easy way to see if it's an access control restriction. Have the network administrator go to the User Profile Management (UPM) services menu and change the user ID type from "user" to "admin." If the user can then gain access, the administrator needs to change the user's security level to allow file access.

Guest User ID

There are some unique characteristics about the guest ID and guest group. Both RIPL and diskboot DLR machines need guest access to the IBMLAN\DOSLAN\NET directory to get started on the network.

If an APPLY has been executed above the NET directory, access to necessary files will be denied. The guest ID or DLR machine trying to start the network then displays a message indicating that the server is not available or cannot copy XSRW.BAT. To correct this error, have the administrator give the guest ID "R" access to the IBMLAN\DOSLAN\NET directory.

An error will also occur if the guest ID has been deleted or was inadvertently assigned a password. This can be resolved by adding back the guest ID or removing the password requirement for the guest ID.

Running Laserjet II and III on a Network

Network printing problems may occur while using the Laserjet II and III printers with the auto-continue function, which is a feature of the printer. While running on a network, these printers should have "Auto Continue" turned *on* even though the default for these models is *off*.

An APPLY Function Tip

The APPLY function should be used with care. This function should never be used on the root drive above the IBMLAN directory. Doing so changes access profiles for directories below the IBMLAN directory. Users would then receive

"access denied" messages when trying to use resources they could access before the APPLY was made. Users may also lose logon assignments. Access profiles are setup for these directories at installation, and any APPLY made afterwards changes profiles. (*Note: An APPLY will not affect any files under the IBM LAN Directory when used with LAN Server 1.3.*)

Common Network Errors and Solutions

Figure 2 shows some LAN Requester/Server error messages and possible solutions.

— Monte Hall, IBM, Austin

Error	Solution
NET3055	Check the cable connecting the PC to the ring
NET3055	Make sure that the sessions and commands parameters in the NetBIOS section of the Communications Manager CFG file are larger than the sessions and commands parameters set on the NET1 line of the IBMLAN.INI file
NET3056	
NET3195	
NET3062	If this error is on an additional server, go through the resync procedure listed at the end of the READ.ME file on diskette 12 of CSD 4098
NET9853	Make sure the machine ID listed in the IBMLAN.INI at the requester is not the same as the user ID

Figure 2: LAN Requester/Server Errors and Possible Solutions

New Products

Hardware

IBM Personal System/2 Model 95 XP 486 (8595-0G9 AND 0GF)

IBM is extending the Personal System/2 (PS/2) Model 95 XP 486 family with two new entry models. These models are based on Micro Channel architecture and use the new i486SX™ (486SX) 20 MHz microprocessor. The 486SX consists of a 32-bit 486 processor without the numeric co-processor unit. This results in an entry 486 processor that performs at speeds comparable to that of a 33 MHz 386 processor. These entry models may be upgraded to the more powerful 486/25 MHz or the 486/33 MHz processors via the IBM PS/2 486/25 and 486/33 Processor Upgrade Options.

The PS/2 Model 95 XP 486 (8595-0G9) is a 486SX/20 MHz system with a 160 MB SCSI fixed disk installed as the standard direct access storage device. The PS/2 Model 95 XP 486 (8595-0GF) is a 486SX/20 MHz system with a 400 MB SCSI fixed disk installed as the standard direct access storage device. In addition, these new models feature 4 MB of memory standard on the system board. The new PS/2 Models 8595-0G9 and 0GF, with the addition of 400 MB fixed-disk file options, can now provide up to 2 GB of internal data storage. This capacity better meets requirements for large data bases typical in local area network (LAN) server and multi-user host applications. All other standard features, such as processor upgradeability, XGA graphics, memory optimization, and expandability attributes of the PS/2 Model 95 XP 486 platform remain unchanged.

Also, a 487SX co-processor is announced, which allows numeric process-

ing capabilities to be added to the 486SX versions of the Models 90 and 95. The IBM PS/2 Math Co-Processor 487SX/20 MHz contains the new i487SX co-processor with installation instructions packaged as an optional feature. The i487SX replaces the i486SX processor and has all of the capabilities of the i486SX, plus integrated floating-point processing to enhance performance for compute-intensive applications.

The operating systems supported by the PS/2 Model 95 XP 486 are OS/2 Standard Edition Versions 1.2 and 1.3, OS/2 Extended Edition Versions 1.2 and 1.3, DOS Versions 3.30 and 4.00, AIX PS/2 1.2.1, and IBM 4680 Operating System Versions 2 and 3.

Highlights:

- New processor complex featuring the 486SX/20 MHz microprocessor
- 4 MB standard parity memory, expandable to 32 MB on the system board
- PS/2 SCSI 32-bit bus master adapter with cache
- Up to 2 GB of internal high-speed data storage
- Eight 32-bit Micro Channel expansion slots (one slot is used for the SCSI adapter and one is used for the XGA Display adapter/A)
- Seven internal storage device bays supporting a combination of 3.5-inch, half-high drives and 5.25-inch, full-high drives
- XGA Display Adapter/A with 512 KB video memory
- One DMA serial port and one DMA parallel port
- Selectable boot and easy-to-upgrade licensed system programs

Letter # 191-059, April 23, 1991

IBM Personal System/2 Model 95 XP 486 (8595-0JF AND 0KF)

IBM is extending the Personal System/2 (PS/2) Model 95 XP 486 family of systems by providing even higher capacity fixed disk storage models. PS/2 Models 8595-0JF and 0KF feature 486/25 MHz and 486/33 MHz 32-bit microprocessors, respectively, with a high-performance, 400 MB SCSI fixed disk installed as the standard direct access storage device. In addition, these new models feature 8 MB of memory standard on the system board. The new PS/2 Models 8595-0JF and 0KF, with the addition of 400 MB fixed disk options, can now provide up to 2 GB of internal data storage. These options allow the the systems to better meet requirements for large data bases typical in local area network (LAN) server and multi-user host applications.

All other technologically advanced, standard features such as processor upgradeability, Extended Graphics Array (XGA) graphics, memory optimization, and the expandability attributes of the PS/2 Model 95 XP 486 platform remain unchanged.

The operating systems supported by the PS/2 Model 95 XP 486 are OS/2 Standard Edition Versions 1.2 and 1.3, OS/2 Extended Edition Versions 1.2 and 1.3, DOS Versions 3.30 and 4.00, AIX PS/2 1.2.1, and IBM 4680 Operating System Versions 2 and 3.

Highlights:

- Processor complex featuring the 80486/25 MHz or 33 MHz microprocessor
- 8 MB standard parity memory, expandable to 32 MB on the system board
- PS/2 SCSI 32-bit bus master adapter with cache

- 400 MB SCSI fixed disk with 11.5 ms average seek time
- Up to 2 GB of internal high-speed data storage
- Eight 32-bit Micro Channel expansion slots (one slot is used for the SCSI adapter, and one is used for the XGA Display Adapter/A)
- Seven internal storage device bays supporting a combination of 3.5-inch, half-high drives and 5.25-inch, full-high drives
- Enhanced Performance XGA Display Adapter/A standard, providing 1024 x 768 resolution
- One DMA serial port and one DMA parallel port
- Selectable boot and easy-to-upgrade licensed system programs

Letter # 191-058, 4/23/91

IBM Personal System/2 Model 90 XP 486 (8590-0G5 AND 0G9)

IBM is extending the Personal System/2 (PS/2) Model 90 XP 486 family with two new entry models. These models are based on Micro Channel architecture and utilize the new i486SX (486SX) 20 MHz microprocessor. The 486SX consists of a 32-bit 486 processor without the numeric co-processor unit. This results in an entry 486 processor that performs at speeds comparable to that of a 33 MHz 386 processor. These entry-level models may be upgraded to the more powerful 486/25 MHz or the 486/33 MHz processors via the IBM PS/2 486/25 and 486/33 Processor Upgrade Options. If the user requires numeric-intensive processing capabilities, this function can be obtained by installing the IBM PS/2 Math Co-Processor 487SX/20 MHz optional feature.

The PS/2 Model 90 XP 486 (8590-0G5) is a 486SX/20 MHz system with a 80 MB SCSI fixed disk drive installed as the standard direct access storage device. The Personal System/2 Model 90 XP 486 (8590-0G9) is a 486SX/20 MHz sys-

tem with a 160 MB SCSI fixed disk drive installed as the standard direct access storage device. In addition, these new models feature 4 MB of memory standard on the system board. The new PS/2 Models 8590-0G5 and 0G9, with the addition of 400 MB fixed disk file options can now provide up to 960 MB of internal data storage to better meet requirements. All other standard features such as processor upgradeability, Extended Graphics Array (XGA) graphics, memory optimization and expandability attributes of the PS/2 Model 90 XP 486 platform remain unchanged.

The operating systems supported by the PS/2 Model 90 XP 486 are OS/2 Standard Edition Versions 1.2 and 1.3, OS/2 Extended Edition Versions 1.2 and 1.3, DOS Versions 3.30 and 4.00, AIX PS/2 1.2.1, and IBM 4680 Operating System Versions 2 and 3.

Highlights:

- New processor complex featuring the 486SX/20 MHz microprocessor
- 4 MB standard parity memory, expandable to 32 MB on the system board
- Enhanced performance XGA graphics integrated on system board providing 1024 x 768 resolution
- Up to 960 MB of internal high-speed data storage
- PS/2 SCSI 32-bit bus master adapter with cache
- Four internal storage device bays supporting three, 3.5-inch half-high drives and one, 5.25-inch half-high drive
- Four 32-bit Micro Channel expansion slots (one is used for the SCSI adapter)
- Two DMA serial ports and one DMA parallel port
- Selectable boot and easy to upgrade licensed system programs

Letter # 191-057, April 23, 1991

IBM Personal System/2 Model 30 286 (8530-E41)

The Personal System/2 (PS/2) Model 30 286 (8530-E41) system unit is a 45 MB fixed disk drive version of the Model 30 286 and has 1 MB 120 ns memory standard on the planar. The Model 30-E41 utilizes the 80286 microprocessor operating at 10 MHz with one wait state to system memory and has the following integrated functions:

- Parallel port
- Serial port
- Pointing device port
- Keyboard port
- 1.44 MB diskette drive support
- VGA graphics

The operating systems supported by the Model 30-E41 are DOS Versions 3.30 and 4.00, OS/2 Standard Edition Versions 1.1, 1.2, and 1.3, and OS/2 Extended Edition Versions 1.1, 1.2 and 1.3.

Highlights:

- 10 MHz 80286 processor
- 80287 Math co-processor socket
- 1 MB of standard memory – 16 MB addressability
- 45 MB fixed disk capacity
- VGA color graphics

Letter # 191-060, April 23, 1991

IBM Personal System/2 486/25 and 486/33 Processor Upgrade Options

The Personal System/2 (PS/2) 486/25 Processor Upgrade Option enhances the announced Personal System/2 Model 90 XP 486 (8590-0G5, -0G9) and Model 95 XP 486 (8595-0G9, -0GF) by providing additional processor growth capability. The PS/2 486/25 Processor Upgrade Option features the 32-bit Intel i486 processor running at 25 MHz on a processor complex designed to upgrade the 486SX/20 MHz or 487SX/20 MHz versions of PS/2 Models 90 and 95. The

486SX/20 MHz or 487SX/20 MHz versions of Models 90 and 95 can also be upgraded by the IBM PS/2 486/33 Processor Upgrade Option. These processor upgrade options significantly enhance the range of processor performance that can now be achieved with the expandable processor concept introduced on the PS/2 Model 90 and 95 XP 486 systems.

The 486/25 and 486/33 Processor Upgrade Options are supported by OS/2 Standard Edition Versions 1.2 and 1.3, OS/2 Extended Edition Versions 1.2 and 1.3, DOS Versions 3.30 and 4.00, AIX PS/2 Version 1.2.1, and the IBM 4680 Operating System Versions 2 and 3.

Highlights:

- 25 MHz and 33 MHz 80486 32-bit microprocessors
- Internal memory cache controller and 8 KB internal memory cache
- Internal floating-point processor unit
- Allows processor upgrade in PS/2 Model 90 XP 486 (8590-0G5, -0G9) and 95 XP 486 (8595-0G9, -0GF) systems
- Well-suited for compute and numeric-intensive applications and for heavy use multi-user, multitasking applications

Letter # 191-052, April 23, 1991

Memory Enhancements for PS/2 Model 80 (8580-081, 161, 321), Model 90 XP (8590-0J5, 0J9, 0KD) and Model 95 XP (8595-0J9, 0JD, 0KD)

IBM is enhancing the standard system memory to 4 MB on PS/2 Model 80 (8580-081, 161, 321) systems. IBM is also enhancing the standard memory to 8 MB on IBM Personal System/2 Model 90 XP (8590-0J5, 0J9, 0KD) systems and Model 95 XP (8595-0J9, 0JD, 0KD) systems.

These memory enhancements are available at no additional charge. With this change, all enhanced systems have dou-

bled the standard system memory to better address application requirements.

Highlights:

- Concurrent execution of more applications with additional 2 MB of memory in PS/2 Model 80 (8580-081, 161, 321)
- Concurrent execution of more applications with additional 4 MB of memory in PS/2 Model 90 XP (8590-0J5, 0J9, 0KD) and Model 95 XP (8595-0J9, 0JD, 0KD)
- No additional cost

Letter # 191-056, April 23, 1991

IBM Personal System/2 SCSI Storage Enclosure and IBM Personal System/2 Card to Option Cable

The Personal System/2 (PS/2) SCSI Storage Enclosure (3510-0V0) is an external enclosure that can accommodate one 3.5-inch or 5.25-inch, half-high Small Computer Systems Interface (SCSI) storage device, such as a SCSI fixed disk drive. The enclosure features a 32-watt universal power supply, a cover key lock assembly, and mounting holes in the base assembly for added security. When this enclosure is equipped with a SCSI storage device, it expands the storage capacity of a PS/2 Micro Channel desktop or floor-standing system. The enclosure adheres to the American National Standard Institute (ANSI) SCSI standard X3.131-1986, which supports up to seven SCSI devices attached to either PS/2 Micro Channel SCSI adapter. Multiple enclosures may be stacked to satisfy storage requirements for local area network (LAN) or other applications that require accessibility to large amounts of data.

The PS/2 Card to Option Cable (#1140, 6451139) replaces the current PS/2 Card to Option Cable (#1041, 6451041). This 60-to-50-pin cable attaches the first external SCSI device to a PS/2 Micro Channel SCSI adapter. The required PS/2 Micro Channel SCSI Adapter Option Interface Terminator, included with the PS/2 Card to Option Cable, is a 50-pin

terminator installed on the last SCSI device to terminate the SCSI bus. This new terminator *must be used* for the PS/2 SCSI Storage Enclosure and for external SCSI devices attached in a "daisy-chain" fashion through the PS/2 Option to Option Cable (#1042, 6451042).

Highlights:

- A versatile package that accommodates a wide range of 3.5-inch and 5.25-inch, half-high SCSI storage devices
- Features include cover key lock assembly, mounting holes in base for bolt down, external audio connection capability, and external SCSI ID selector switches

Letter # 191-053, April 23, 1991

Enhanced M-Motion Video Adapter/A for the IBM Personal System/2

The M-Motion Video Adapter/A (#1991, 95F1091) is a display adapter for Personal System/2 (PS/2) system units with Micro Channel architecture. With this adapter, full-motion interactive color video, and VGA graphics and text can be displayed on a standard PS/2 color display. The M-Motion Video Adapter/A also provides full-line audio input/output capabilities with limited quality digital record/playback. Composite Sync (CS) output is available to synchronize up to three compatible video input sources at the same time. The new M-Motion Video Adapter/A (#1991, 95F1091) replaces the original M-Motion Video Adapter/A (#3487, 34F3087).

M-Motion Video Adapter/A software and an installation software diskette provide interactive diagnostics. Composite Sync loop test is available through advanced diagnostics. The installation software is compatible with DOS and OS/2 operating systems. M-Control Program/2, Audio Visual Connection (AVC) Version 1.03, Storyboard™ Live!, and LinkWay™ are supported by the M-Motion Video Adapter/A.

Highlights:

- Increases user performance while reducing rework
- Improves availability and access to data
- Enables personal computing
- Saves personnel training time through reduction in complexity
- Supports business objectives
- Improves productivity
- Manages rate of change
- Supports application development

Letter # 191-037, April 2, 1991

Software**IBM Hollywood Version 1.0**

Hollywood™ is a presentation graphics software product to help presenters in business, government, and education create star-quality presentations with ease. Hollywood includes powerful text capabilities and extensive charting tools that can create a wide variety of data charts, such as bullet, pie, bar, tables, and organization charts. In addition, a complete set of drawing tools is available for annotating charts and creating diagrams.

Hollywood is planned as a family of presentation graphics products. The Windows 3.0 version of Hollywood was announced on May 7, 1991. In addition, IBM intends to provide a version of Hollywood for the OS/2 Presentation Manager environment.

Highlights:

- Creates complete presentations: transparencies, slides, paper, handout notes, speaker notes, and on-screen presentations
- Creates data, graph, and word charts in most forms
- Creates a presentation from an outline
- Has drawing capabilities
- Works with most graphic file formats

Letter # 291-214, May 7, 1991

IBM Stories and More

Stories and More includes interactive, mouse-driven activities, which extend an early reader's experience with literature. The Teaching and Learning with Computers (TLC) teacher support materials accompanying this product extends IBM's Integrated Language Arts TLC approach to computer use in the classroom.

Highlights:

- Literature-based reading approach helps foster a sense of excitement for stories and helps build an appreciation for the power of language
- Engaging reading voices bring the literature alive, giving greater meaning to the text of the stories
- Interactive mouse provides alternative to keyboard and enhances student interest
- Story activities build reading comprehension by asking students to recall, analyze, and interpret what they read, and then compare the new information to what they already know

Letter # 291-216, May 7, 1991

IBM Touch Typing for Beginners Version 2.00

IBM Touch Typing for Beginners Version 2.00 is designed to teach correct finger placement and touch typing, provide practice for all letters, numerals and most function keys, and help improve typing speed and accuracy. The program is recommended for second grade and above, but may be used by anyone wishing to improve basic typing skills. Colorful animated graphics make full use of PS/2 capabilities. The program runs stand-alone or in a networked environment.

The program supports a commitment to the Teaching and Learning with Computers (TLC) approach for computer use in the classroom while giving students keyboarding skills required by Writing to Read, Writing to Write, and all IBM Basic Skills courseware.

Highlights:

- Improves basic typing skills through the use of motivating games, beautiful graphics, and an easy-to-use menu system
- Enhances learning through a self-paced, sequential lesson design
- Provides teachers with customization capabilities
- Enriches the product offerings supported by the Teaching and Learning with Computers integrated approach
- Functions on all models of PS/2s in a stand-alone or network environment
- Appropriate for a wide range of student abilities

Letter # 291-137, April 2, 1991

Index to Past Issues of *IBM Personal Systems Technical Solutions*

Issue 2, 1991 (G325-5011)

IBM PS/2 Model 90 XP 486 and Model 95 XP 486
Choosing an I/O Bus Architecture
The Network Is the Message
Invoking Printer Job Properties
Comparing PC-DOS, OS/2, and AIX PS/2
Programming PM Using the COBOL/2 Bindings
Installing and Using the DOS Database Requester
OS/2 LAN Server 1.3 Overview
IBM Windows Connection 2.0
SNA Definitions for 3270 Emulators – Part II
IBM THINKable

Issue 1, 1991 (G325-5010)

XGA – Raising Video Expectations
Choosing betw. Shielded and Unshielded Wiring for Data Transmission
Compatibility of LAN Servers and Requesters
Running DOS LAN Requester and Novell Netware Concurrently
Breaking the 640 KB DOS Memory Barrier
Understanding an OS/2 CONFIG.SYS File
OS/2 EE 1.2 Database Manager Performance
OS/2 EE 1.2 Competitive Performance
An Intelligent Front-End EASEL Application
Enabling Software for National Language Support
SNA Definitions for 3270 Emulators
Diskette Failures Caused by Contamination

Issue 4, 1990 (G325-5009)

First Look at the New IBM PS/1 Computer
Using the IBM 4019 LaserPrinter Effectively
Micro Channel Interface Chip Sets
Token Ring Bus Master LAN Adapters
Extension of Wiring Rules for 4-Mbit/s Token Ring Using UTP Lobes
SCSI and DISK.386.SYS
Operating System Platforms: A Business Perspective
Minimum OS/2 1.2 DASD Requirements
User Profile Management
Understanding OS/2 1.2 LAN Server Performance
PM: An Object-Oriented Approach
DOS 4.00 SHARE
A “C” Programming Model for DOS Device Drivers
An Electronic Bulletin Board for PC Users

Issue 3, 1990 (G325-5007)

DOS – A Look under the Hood to See How It Spins
Memory Management in a DOS Environment
FASTOPEN – The DOS Performance Enhancer
DOS 4.00 Compatibility Issues
'Out of Environment Space' Errors
A New LAN Requester for DOS Systems
Creating a Dialog Box Dynamically Using WinCreateDlg
An Alternative for the OS/2 START Command
CUA: A Consistent Interface

Issue 2, 1990 (G325-5006)

OS/2 End User Advantages
What's New in OS/2 Standard Edition Version 1.2?
An Application Developer's View of OS/2
Object-Oriented Programming with C and OS/2 PM – Is It Possible?
Design Goals and Implementation of the New HPFS
OS/2 EE 1.2 Database Manager – Remote Data Services
OS/2 EE Database Manager Precompiler API
UNION, INTERSECT, EXCEPT
Writing a Database Manager COBOL/2 Program
Database Manager Programming with Procedures Language 2/REXX
APPC Performance Tips for OS/2 EE
EASEL OS/2 EE PROFS: Host Code Interface
PS/2 RPG II Application Platform and Toolkit
The IBM Independence Series Products

Issue 1, 1990 (G325-5005)

Introduction to Local Area Networks
IEEE 802.3 LAN Considerations
The IBM Token-Ring Network: A New Generation
How to Design and Build a 4-Mbit/s Token-Ring LAN
How to Design and Build a 16-Mbit/s Token-Ring LAN
Making the Cabling Decision
IBM Cabling System Highlights
Communication Strategy for Growth

Issue 4, 1989 (G325-5004)

Requirements for Advanced Bus Architecture
Micro Channel System Configuration Considerations
Features and Benefits
Bus Masters and Applications
Overview of Extended Micro Channel Functions
New Micro Channel Features
SCB – An Architecture for Micro Channel Bus Masters
Design Alternatives with Micro Channel Systems
The IBM PS/2 Micro Channel SCSI Adapters
PS/2 Wizard Adapter
Experience in Bus Master Design
Bus Master Adapters from Independent Option Vendors
Bus Masters and OS/2
Micro Channel Issues in AIX PS/2
Information for Developers
Book Report: “The Winn Rosch Hardware Bible”

“ Suspend and resume adds convenience to the L40 SX Laptop. (page 6)

“ The OS/2 2.0 shell has been redesigned to give users a single interface to manage multiple types of objects. (page 10)

“ When porting applications to either OS/2 or AIX PS/2 from DOS, all system calls must be recoded. (page 31)

“ With dual displays, users can focus on the PM screen earlier than normal when switching back to the PM session. (page 42)

“ Today, a personal computer puts more power on your desk than resided in the “glass house” of the 1960s. (page 43)

“ Throughout our time in Arizona, the STS and BUSMON PS/2s were attached with duct tape to a missile launcher, which was on the side of the helicopter. (page 48)

“ DCAF adds a new dimension – electronic monitoring from anywhere on the network. (page 58)

“ IBM’s STP cables can support high data rate transmissions because of the shield that protects the environment from electromagnetic interference. (page 59)

“ The IBMLAN.INI BIGBUFS provide a way to control memory utilization. (page 64)

“ Basically, seven different protocols are used for communication and packet routing within the native NetWare environment. (page 68)

G325-5012-00

